HRL-TR-90-36

FORCE

HUMAN RESOURCES

LABORATORY

TRAINING TACTICAL DECISION-MAKING SKILLS: AN EMERGING TECHNOLOGY

Fritz H. Brecke

Logicon, Incorporated
Tactical and Training Systems Division
4010 Sorrento Valley Boulevard
P.O. Box 85158
San Diego, California 92138-5158

Michael J. Young

LOGISTICS AND HUMAN FACTORS DIVISION
Wright-Patterson Air Force Base, Ohio 45433-6503

October 1990

Final Technical Report for Period November 1986 - April 1990

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.

BERTRAM W. CREAM, Technical Director
Logistics and Human Factors Division

JAMES C. CLARK, Colonel, USAF
Chief, Logistics and Human Factors Division

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1990	Final Report - November 1986 - April 1990

4. TITLE AND SUBTITLE
Training Tactical Decision-Making Skills: An Emerging Technology

5. FUNDING NUMBERS
C - F33615-88-C-0020
PE - 62205F
PR - 3017
TA - 08
WU - 15

6. AUTHOR(S)
Fritz H. Bracke
Michael J. Young

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Logicon, Incorporated
4010 Sorrento Valley Boulevard, P.O. Box 85158
San Diego, California 92138-5158

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)
Logistics and Human Factors Division
Air Force Human Resources Laboratory
Wright-Patterson Air Force Base, Ohio 45433-6503

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFHRL-TR-90-36

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Training in decision-making skills directly contributes to the combat readiness of Battle Staff Officers assigned to Tactical Command and Control positions. More affordable, accessible and effective training technologies are required to supplement the Command Post Exercises currently in use. This is the final report for a 3-year research and development (R&D) effort aimed toward the development of such technologies. The R&D problem was viewed from the perspective of Instructional Systems Development (ISD) as one of defining instructional content and strategy. The analysis, design, and development phases of an ISD process were completed for a specific, representative decision-making task performed by Fighter Duty Officers of an Air Support Operations Center (ASOC). The project resulted in the definition of a generic instructional strategy for practice of a class of decision-making tasks, and two computer-based systems called KEATS and SuperKEATS. KEATS was developed in Smalltalk and runs on PC/AT compatibles. SuperKEATS was developed in KEE and runs on a Sun 3/60 workstation. KEATS' primary function is to support the acquisition of decision task knowledge which is used as instructional content. SuperKEATS provides a practice environment for decision-making skills, with a full range of natural and artificial feedback. Only limited evaluations were performed, but the results were generally positive. Further evaluations and continued development are recommended.

14. SUBJECT TERMS

computer-based training	instructional strategy	micro-computer
decision-making training	knowledge acquisition	skill acquisition
expert knowledge	knowledge engineering	tactical command and control

15. NUMBER OF PAGES
78

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

SUMMARY

Battle managers assigned to Air Force Tactical Air Control Systems need to be competent decision-makers. improved technologies are required to develop and deliver training in decision-making skills at an affordable cost. Training should be available at peacetime duty stations, run on microcomputers available at the squadron level, and not require instructors.

A 3-year research and development project, sponsored by the Ground Operations Branch of the Air Force Human Resources Laboratory, and performed by Logicon's Tactical and Training Systems Division, was conducted to develop technologies that would make training for decision-making skills more affordable, more accessible and more effective.

The project in its entirety was viewed as an Instructional Systems Development challenge, where instructional content and strategy for decision training objectives had to be specified. Two proof-of-concept systems were developed during this project. The first prototype, Knowledge Engineering and Training System (KEATS), a computer-based (Smalltalk and PC/AT compatible) system, was applied to the Fighter Duty Officer position at an Air Support Operations Center and helped identify schemas for situation assessment and planning, and decision rules for tasking of air requests. KEATS was also designed as a practice environment for decision-making, but could not support free play. The second prototype, SuperKEATS, was designed strictly for training purposes and does support free play in decision-making scenarios of adjustable complexity. The methodological and technological products of the project are discussed. Based on initial results, continued development of the work begun with this project is recommended.

PREFACE

With its establishment in 1980, the Logistics and Human Factors Division's Ground Operations Branch, a part of the Air Force Human Resources Laboratory (AFHRL/LRG) at Wright-Patterson AFB, Ohio, began a comprehensive program to develop advanced technology to improve training for USAF Tactical Command and Control (C^2) Battle Staff personnel.

This is the final report for a 3-year R&D project involving the development of technologies capable of providing more accessible, affordable and effective training for decision-making skills. The project was guided by Mr. Michael J. Young of AFHRL/LRG and executed by Logicon's Tactical and Training Systems Division in San Diego, California, under the direction of Fritz H. Brecke. Logicon team members include: Patrick Hays, Donald Johnston, Gail Slemon, Jane McGarvey, and Susan Peters. Particularly heartfelt thanks go to the Air Force subject-matter experts: Lt Col Fred Wilson, Lt Col Lynn Weber, Lt Col Matt Szczepanek, Maj Ken Dekay and Cptn Osborn. The project would not have been possible without them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. PROBLEM

Effective utilization of tactical combat assets depends to a very large extent on the cognitive skills of personnel who assign combat assets to objectives or targets. These decision-making functions are generally performed by "battle managers"; i.e., officers assigned to Tactical Command and Control (TC^2) systems. Rapid, accurate decision-making is a critical combat skill for these officers.

Decision-making skills are expected to be acquired during various types of small- and large-scale exercises which occupy most of the peacetime duty time (over 80 percent) of officers assigned to the Air Force Tactical Air Control System (TACS). Brecke, Jacobs, and Krebs found that small-scale exercises focus primarily on procedural and mechanical skills and that the only currently available training opportunities for complex TC^2 decision-making skills are large-scale exercises.

Large-scale exercises can involve many TACS nodes or an entire TACS network. These exercises are of two types: Field Training Exercises (FTXs), which involve actual troop and aircraft movements; or Command Post Exercises (CPXs), which involve simulated movements. Both types of large-scale exercises require lengthy and manpower-intensive planning efforts and, during execution, consume high amounts of expensive resources. CPXs may offer some planning and cost advantages over FTXs in that actual movements of operational assets are substituted by computer simulations. However, CPXs can require large numbers of "supporting players" who simulate the responses of larger units. Nevertheless, only few exercises of either type can be staged per year and even these few training opportunities cannot all be attended by all battle staff personnel who would benefit from them. Training in decision-making skills is therefore currently both "hard to get" and extremely costly.

The effectiveness and efficiency of training provided by large-scale exercises are not easily measurable. However, it is doubtful that the sporadic training opportunities afforded by these exercises can lead to the rapid achievement and continuous maintenance of high proficiency levels in decision-making that are required by combat-ready corps of battle managers. The effectiveness and efficiency of exercises as training vehicles are further eroded by lack of explicit feedback on decision-making performance; by the inability to tailor exercises to individual training needs; and by the apparently inevitable "break-in" period at the beginning of each exercise, where participants are preoccupied with the mechanical and procedural problems of learning how to "play the game" rather than with solving tactical decision problems.

These shortfalls in current training capability for decision-making skills create a need for training that is more affordable, accessible, and effective than that now provided by large-scale exercises. Ideally, training in decision-making skills should be available at a battle manager's peacetime duty station; it

should require no more resources than those affordable at a squadron or wing level; and it should reliably produce combat-qualified decision-makers within reasonably short training times.

Developing a training technology to meet these requirements involves more than merely offloading current large computer system tactical battle simulation technologies to microcomputers or waiting for microcomputer technology to reach the required levels of computing power and storage capacity. Current C^2 simulation technology is manpower-intensive and thus impractical at the unit level. Means must be found to replace "supporting" participants with "intelligent simulated" participants and to reduce exercise preparation costs through well-designed exercise editing packages. Also, a means must be found to model a reasonably intelligent and hostile opponent who follows doctrines and procedures that are "alien" (i.e., different from ours). All of this is becoming increasingly feasible: The modeling problem is becoming easier to solve with modern programming techniques and languages, and hardware that is powerful enough to run such models is now available.

Developing decision training technology is not a task that can be accomplished by computer engineering alone, however; it takes "cognitive engineering" (Rasmussen, 1986) as well. Montague expressed this notion as follows: "The analysis of competent performance and its development that must be done to plan instruction must include cognitive organization and structures, and attend to the phases and processes involved in acquisition" (Montague, 1986, p. 5). In more general terms, any attempt at developing improved training and/or training design technologies should aim to translate a theory of skill performance and acquisition into common practice.

The project reported here was launched to investigate the application of cognitive psychology and cognitive science to decision-making training. Since Nickerson and Feehrer's (1975) landmark report, an entire new body of knowledge has emerged that deals with the nature and acquisition of expertise. This work, much of which was sparked by research and development (R&D) on differences between novices and experts, has prompted a keen appreciation of the role of domain knowledge in expertise and has spawned powerful new concepts and theories dealing with the representation of knowledge and its acquisition. If it is possible to synthesize these new approaches to the extent that prescriptive instructional design principles can be formulated, then more effective solutions to the decision training problem may be found.

The need for more affordable and effective decision training technology has been recognized and R&D efforts to produce such technologies have been accomplished. Wilson (1982) wrote a master's thesis at the Naval Postgraduate School describing an "Interactive Micro-Computer Wargame for an Air Battle," which ran on an Apple III and was written in UCSD Pascal. Obermayer, Johnston, Slemon, and

Hicklin (1984) reported the development of a micro-based (WICAT System 150) multi-user system which simulated a simple Anti-Submarine Warfare (ASW) scenario. The system served as a research prototype; it required player decision inputs and featured one artificial, knowledge-based team member. Madni, Ahlers, and Chu (1987) demonstrated the modeling of an intelligent opponent in a knowledge-based simulation prototype running on a Symbolics 3670 workstation. The prototype was designed to train Tactical Action Officers in the kind of decision-making skills needed during naval surface warfare. The Army Research Institute for the Behavioral and Social Sciences has focused on experimental Computer-Aided Instruction (CAI) in the training of decision-making skills for armor officers. One result of this work is the "Armor Tactical Concepts Tutor (ARTACT)," which runs on the Army's microcomputer-based Electronic Information Delivery System (EIDS). Stoddard, Kern, and Emerson (1986) announced the development of a cognitive skills tutor which continues and expands the ARTACT work.

The present project builds on these R&D efforts and continues a line of research that was begun by AFHRL/LRG in 1984 with the so-called Combat Planning and Attack Capability (COMPAC) study (Krebs et al., 1984), a comprehensive survey of factors impacting combat readiness of battle managers in the Air Force's TC^2 system. One of the conclusions of that study was that both formal schoolhouse training and training provided by exercises could and should be improved. The first steps toward such improvements were initiated in a follow-on project designed to develop improved methods for training requirements analysis, training design and training development of battle manager wartime skills. That study analyzed the existing training situation in greater detail and showed that the traditional Instructional Systems Design (ISD) procedures were fully adequate for training aimed at procedural and/or mechanical skills, but were ineffective for higher-level cognitive skills such as decision-making skills. New methods for developing task lists and defining wartime training requirements for battle managers were developed and, to some extent, evaluated. These new methods proved more effective and efficient in developing accurate and complete task lists for battle manager positions, but failed to produce detailed task breakdowns for complex emergent tasks such as decision-making tasks.

The overall goal of the project reported here was to go beyond training requirements analysis and to produce prototype training packages for specific decision-making tasks. The prototype training packages should reliably result in the acquisition of a targeted decision-making skill; should be presentable by means of microcomputer media without a human instructor; and should, to the extent feasible, be instantiations of instructional principles derivable from the existing base of instructional theory and the current understanding of the performance and acquisition of complex cognitive skills.

The results of the preceding study indicated that the achievement of this project goal was dependent on being able to describe explicitly how a given decision-making task is performed. That study had shown that the knowledge of "how to" perform a decision-making task was not available in any documented form and that neither the traditional nor the alternate methods developed during the study were effective in externalizing this task knowledge.

The overall goal of the present effort was therefore subdivided into two specific objectives:

Objective 1:

Develop a methodology to externalize Subject-Matter Expert (SME) task knowledge for decision-making tasks.

Objective 2:

Develop training methodologies which reliably facilitate achievement of high performance in decision-making tasks.

The present report is the final report for the 3-year effort undertaken to achieve these objectives. The effort was performed in three phases. Reports were written for each of the three phases. The reports on Phases II and III (Brecke et al., 1989 and Brecke et al., 1990) were published and provide details on the development of two prototype systems that were developed. This final report provides an overview of the entire effort.

The remainder of this report is divided into two sections. The next section describes the work accomplished in each of the three project phases (in less detail than the Phase II and Phase III reports). The third and final section presents a discussion of the methodologies developed during this project, their technical merits, and their cost effectiveness.

II. PROJECT PHASES

During Phase I, the basic approach and the study domain were selected. Phase II was devoted to work on developing a methodology to externalize task knowledge. Phase III was devoted to work on developing training methodologies. This section describes the work performed and the results achieved in each of the three project phases.

Phase I: Preparation

Phase I laid the necessary groundwork for the remainder of the project. Two activities were particularly significant in terms of shaping the work in the follow-on phases: the development of the technical

approach and the selection of the target study domain. The results of these activities (i.e., the approach and the selected domain) are described below.

Approach

The project in its entirety was viewed as an Instructional Systems Design (ISD) problem, where a generic training design solution for a class of skills (decision-making) is sought. An ISD problem exists if some of the variables that define an instructional event or product are "given" and others are "unknown" or "to be determined." Merrill and Wood (1974) identified four "relatively independent" facets of instruction representing sets of instructional variables: Learner Aptitude, Content (or subject matter), Instructional Strategy, and Delivery System. Frank (1969), in a more formal cybernetic theory of instruction, used the same dimensions and added two more: the Purpose (or Objective) of Instruction and the Environment within which instruction is to take place.

The ISD problem to be solved by this project could thus be specified as one in which four dimensions (in Frank's terms) were given and two were sought. The following dimensions were given or predetermined:

Training Objective	Specified as a class of skills to be trained: Decision-Making Skills.
Learners	Battle Staff Officers assigned to Tactical Command and Control Systems.
Delivery System	Microcomputer media.
Environment	In-house environment at Tactical Command and Control Units.

Solving the problem involved defining the two remaining dimensions which are required to completely specify an instructional event:

Content	Classes and instances of domain knowledge.
Instructional Strategy	Types of instructional actions and their sequencing.

A generic instructional strategy for training decision-making skills either is available (in ISD manuals) or should be specifiable from a theoretical and empirical base. The instructional content is the domain knowledge employed by battle managers during the performance of decision-making tasks. This knowledge does not exist in explicit form (in manuals or regulations).

Given instructional content in explicit form and a defined instructional strategy, prototype training systems can be specified, designed, and developed. The prototype systems must be specific instantiations of the instructional strategy, use the acquired instructional content, and reflect the implementation constraints imposed by the features and limitations of extant microcomputer hardware and software.

Ideally, the prototypes must then be subjected to cycles of formative evaluation trials and revisions. The eventual product is a validated instructional delivery method and a set of guidelines or "instructional design heuristics" (Montague, 1986) for application to other decision domains.

Prior to obtaining the instructional content (in Phase II), two possible approaches for a prototype training system for decision-making were actively considered. One approach would put the trainee in a specific situation requiring a decision, and then give feedback based on that decision. This would require an expert system to evaluate the student response and would resemble an intelligent tutoring system (ITS) in scope. Instructional strategies could vary from an exploratory mode, where the student could set up his/her own situations, to a closely guided sequence of situations starting, for example, with a simple epitome. The second approach would start the trainee with a specific situation, but require a series of decisions over time as the environment changed, situations evolved, and effects of earlier decisions became apparent. This *simulation* approach would require modeling of objects in the study environment. Training strategies in this case would be dependent on starting situations (which could begin with a simple epitome and become more complex) and the types of feedback available to the trainee.

Initially, the ITS approach was favored for reasons of training and feasibility. Using an ITS approach, the training designer could ensure that the trainee had been exposed to a complete set of situations, whereas with a simulation approach, the student might never have to face certain obscure yet critical types of decisions (e.g., by adopting a particular game strategy). Using an exploratory mode, the student could set up "what if" situations to examine a range of possibilities; however, this would not be possible with a simulation. In addition, there was considerable doubt as to the feasibility of developing a reasonable simulation environment with the limited resources available (time, number of people, computers). A simulation environment that was unrealistic would not be accepted by the end users; an environment that was oversimplified would not produce operational training.

On the other hand, a simulation approach offered certain advantages, the most obvious being "natural" feedback over time. With a simulation, the user would have the opportunity to see the effects of decisions and learn how to recover from less-than-optimal choices. Neither of these would be effectively practiced with an ITS approach. Additionally, there was a question of "brittleness." To evaluate and give feedback on the range of decisions available in an ITS, a complete expert model would have to be developed. Because weapon systems and tactics (both ours and those of our adversaries) are continually changing, the specific knowledge in the model would, unless maintained, become brittle and fail to properly evaluate new situations. With a simulation approach, the expert model requirements could be reduced and the prototype training system would hopefully be less brittle.

The choice between these approaches was not resolved during Phase I, but required knowledge of the study domain and instructional content obtained in Phase II. A key finding was that in the study domain a considerable amount of background detail must be known before a decision can be made. This typically requires that the expert have up to an hour to become familiar with the situation. In the field, this is done via briefings first thing in the morning, followed by reports, orders, and requests during the day. This long setup time would make the closely guided ITS approach slow and ineffective because the situation would change between each decision and require learning anew. The exploratory ITS approach could minimize this problem, however, because the user could change a few situation parameters at a time and not have to relearn the entire scenario.

Study Domain

The general target domain for the project was the Tactical Command and Control System of the Air Force. Within the general domain of Air Force TC², the focus was on a specific type of node called the Air Support Operations Center (ASOC). The ASOC is an Air Force TC² unit which is "assigned" to and co-located with an Army Corps Headquarters (see Figure 1). Its basic role is to supply air support missions in response to demands originating from the Army Corps' front-line divisions. The ASOC thus has to solve a problem which, in general terms, can be viewed as a supply management problem. The problem is nontrivial because the demand for air support may exceed the supply of available aircraft, and both supply and demand vary independently over time in ways that are only partially predictable.

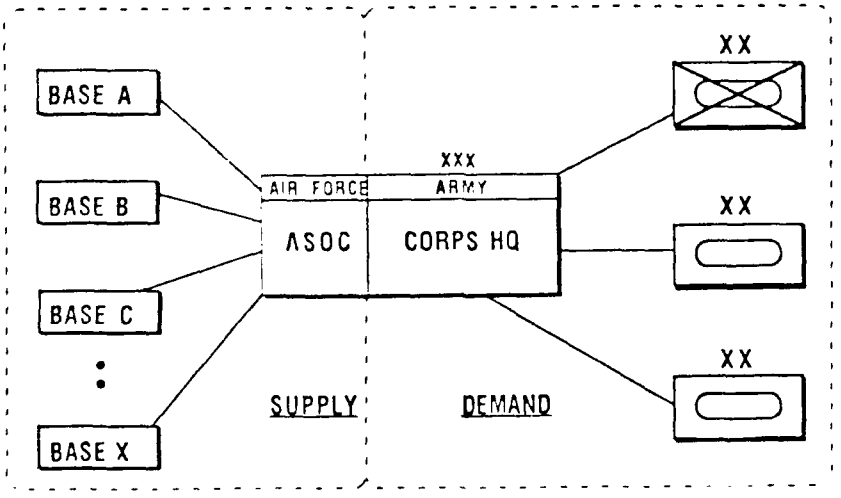


Figure 1. Project Study Domain: Air Support Operations Center (ASOC).

Planning and Operations at the ASOC. The ASOC's method for solving this supply management problem is a combination of advance planning and subsequent plan execution, with ad hoc adjustments. Advance planning is performed during the night shift when little or no combat action is occurring.[1] Plan execution, usually referred to as "operations," occurs during the daylight hours when most tactical combat action occurs.

The plan produced by the night shift crew is basically a schedule which maps predicted supplies of available sorties into anticipated demands for air support for each of the corps' front-line divisions. This plan or schedule covers the daylight hours; i.e., the time from first light to last light. Information for the supply side of the plan comes primarily from the Air Force Air Tasking Order (ATO) and Daily Operations Order (DOO). Information for the demand side of the plan comes primarily from the Army Corps' plans and from intelligence sources. The Army has the prerogative of determining the "priority of fire," which is defined in percentage allocations of the total available sorties to each of the front-line divisions. The ASOC determines the details of timeframes, squadron assignments, and tasking modes.

Plan execution is essentially the responsibility of the day shift crew. The day crew uses the night shift's plan as a baseline and modifies it to take into account the most recent information regarding available assets and battlefield situation. The plan is adjusted and readjusted throughout the day in response to changes in the tactical situation and as preparation for anticipated developments in the near future. The day crew therefore deals both with the present and with the near future, and with actual and anticipated supply and demand problems. The key to successful air support service from the ASOC to the Army Corps is to manage sortie generation and sortie expenditure such that the Corps receives optimal support in the present, throughout the day, and over the foreseeable length of a conflict.

The differing roles of the night and day crews at the ASOC reflect a partition into Planning and Operations (plan execution) that is common throughout the Air Force TC^2 System. The ASOC therefore can be considered a representative subdomain of the offensive side of tactical command and control.

Within the ASOC domain, this project focused on the decision problems faced by the day shift (i.e., operations personnel). The decision problems involved in planning have already been investigated in detail through projects like the Tactical Air Operations Team Training System (TAOTTS) (Barnthouse, 1989) and the Tactical Expert Mission Planner (TEMPLAR) (McCune, 1985; Priest, 1986). TAOTTS and TEMPLAR provide training and decision support to the process of developing Air Tasking Orders.

[1] This reprieve may soon disappear with the advent of technologies that enable air-to-surface combat at night and under other very low visibility conditions.

It is important to note that the decision problems faced by planners and operations personnel are quite dissimilar. One important distinction is the time available for solving decision problems. Planners usually have much more time than do operations personnel. Planners generally concern themselves only with a relatively distant future; operations personnel must alternate between the present and the near future. Planners decide on a global mapping of sorties into demands, not the details of mission implementation and coordination. Operations personnel must deal with very specific assignments of sorties to very specific requests and they have to deal with the details of mission implementation and coordination.

Several specific examples of the types of decision problems that occur during ASOC operations are given below. They illustrate the type of decision-making skills that need to be trained.

Example 1:

The ASOC Fighter Duty Officer I (FDO 1) receives an air request to neutralize eight tanks within the next hour. The request has been verified by the appropriate officers. The FDO 1 must now decide which of the available airplane assets should be tasked to satisfy this request. If he has no assets to task, he must refuse the request. If weather or threats make mission success questionable, he has to weigh risks against benefits.

Example 2:

The ASOC receives a Corps request to neutralize a large number of tanks attacking along a highway. This type of request generally requires tasking of a fairly large number of airplanes, some of which will be the actual attackers and some of which will fly support missions. This type of request may include participating Army assets. This sort of decision problem is usually handled by the ASOC Director, who decides on force composition, timing, coordination, and other relevant factors.

Example 3:

The ASOC receives weather information that one of the bases where it has airplane assets will be under zero visibility and zero ceiling conditions within an hour. The weather is estimated to clear within 2 hours. The ASOC must now decide how to compensate for the shortfall of available sorties. This may have to involve a complete replanning effort.

Phase II: Acquisition of Task Knowledge

Problem

Making decision task knowledge explicit is difficult. An earlier effort showed that conventional ISD task analysis methods work well for procedural skills but do not capture the detailed task knowledge employed in higher-level cognitive tasks. An alternate technique, developed and employed by the same researchers, approached the problem by eliciting cognitive maps as representations of task knowledge. This technique, though much more efficient than conventional task analysis methods, was not fully effective in capturing task knowledge for complex cognitive tasks. The approach chosen in the present project was based on the idea of applying knowledge engineering techniques used in expert systems development to the problem of acquiring cognitive task knowledge or instructional content.

Unaided Knowledge Acquisition

The specific knowledge and heuristics that enable ASOC personnel to make the kinds of decisions illustrated above represent the domain knowledge that must be made explicit in order to provide the substance needed to train these skills in others. The methods, results, and problems noted during an initial knowledge acquisition effort are described below.

Methods. The plan in exploring the ASOC domain was to go "breadth first" and then to narrow the focus and explore in-depth. This strategy was predicated on the assumption that even the relatively narrow subdomain of ASOC operations was still too broad to permit adequate coverage within the constraints of the project. To further narrow the scope, it was necessary to first achieve a broad (but fairly shallow) understanding of the ASOC domain.

The techniques used during initial knowledge elicitation consisted of focused and structured interviews (Schraagen, 1986). Both types of interviews are driven by some type of agenda. The difference is that focused interviews use a "breadth first" strategy, whereas structured interviews employ a "depth first" strategy.

One or more knowledge engineers and one or more experts participated in each of the interviews. The interviews were tape recorded. The tapes were transcribed and each of the resulting scripts was analyzed by three knowledge engineers in succession. Analysis consisted of identifying knowledge elements within the often rather colloquial discourse. The knowledge elements were transferred to 3- by 5-inch cards, which were then sorted and grouped under topics that emerged during the sorting process.

The intent was to identify and use naturally occurring knowledge categories and not to impose any preconceived classification.

Results. A quantitative description of the initial knowledge engineering activities is presented in Table 1. A total of four knowledge engineering sessions were held in roughly 4 months. Knowledge engineers and experts spent 56 hours together. These sessions produced 26 tapes and 693 pages of transcripts. Analysis of the transcripts produced 1,599 knowledge element cards. The knowledge elements had an estimated redundancy of 40 percent to 50 percent. The analysis process was labor-intensive. Productivity was estimated at about 1.5 knowledge elements per analyst hour.

The entire initial knowledge engineering effort took more than 500 hours of analyst time and produced a broad, uneven, and relatively shallow surface layer of knowledge about the ASOC domain. The analysts involved perceived the effort as inefficient and felt that there were two reasons for this inefficiency: the team's inexperience, and shortcomings inherent in the interview methods. The team had to learn how to elicit knowledge. This detracted at least initially from the productivity of the knowledge engineering sessions. Both interview methods are relatively inefficient, but focused interviews produce much more "noise" (rambling, anecdotal discourse) than do structured interviews. The structured interviews were case-based, meaning that detailed and specific tactical situations were presented to the experts. It was much easier to stay on the target topic with this type of interview, although productivity in terms of knowledge elements was not higher (see Table 1).

Table 1. Unaided Knowledge Acquisition Activities

No.	SMEs	KEs	Contact Hours	Focus and Methods	No. of Tapes	No. of Pages	No. of Cards	No. of Analyst Hours
1	2	1	16	Introductory background in ASOC Focused interview	6	156	604	120
2	1	1	16	ASOC communications, job positions, information flow Focused interview	7	182	361	150
3	1	3	8	Constructing "special packages" Structured interview	7	233	384	160
4	3	2	16	Handling of ATRs, effect of weather, troop movements Structured interview	6	122	250	120
Totals	7	7	56		26	693	1,599	550

KEs = KNOWLEDGE ENGINEERS
SMEs = SUBJECT MATTER EXPERTS

Case-based interview methods did have one significant drawback: The "case," which consisted of a script of events in a defined TC^2 environment, would fall apart if the expert made a decision at some point that would make the subsequent events in the script improbable or impossible. This type of failure indicates the need for some minimal capability to adjust the interview script to unexpected expert reactions. The fallibility of the knowledge engineering script is one that plagues all pre-scripted exercises. As mentioned previously in Section I, recovery from such deviations in exercises is a very labor-intensive process which has to take place during a time-out or at night.

The inefficiency of the knowledge engineering process is well known as the "bottleneck problem" of expert systems technology (Hayes-Roth & Waterman, 1984). In addition to the inherent inefficiency of current methodologies, a number of other problems were identified which inevitably accompany the process and which, if left unattended, can jeopardize the success of a knowledge engineering effort. These problems are described in some detail below.

Knowledge Engineering Problems. Five types of problems were identified by the team. None of these are new or avoidable; they have all been noted and described before by other researchers who have engaged in knowledge acquisition efforts. The problems have been referred to by a variety of names; in the present report, they are referred to as follows:

1. The Uncertainty Problem
2. The Expert Paradox Problem
3. The CATCH-22 Problem
4. The Access Problem
5. The Quid-Pro-Quo Problem.

The UNCERTAINTY problem refers to the fact that domain knowledge is often rule-of-thumb, nondeterministic, and of unknown optimality. Tactical decision problems often have multiple feasible and acceptable solutions. The solutions may differ in "quality," but it is quite difficult to find generic metrics to assess solution quality. Experts may display a low degree of consensus both as to which of several feasible solutions is best and as to the criteria that should be used to determine the best solution.

It is extremely difficult to identify optimal solutions in the tactical environment. There are many reasons for this. In addition to individual differences, there are many factors which are unknown, unpredictable, and fuzzy. A particular solution may optimize effectiveness in a local instance but cripple overall effectiveness. Knowledge engineering in the tactical environment therefore should seek to identify a set of

plausible and reasonable solutions and define their relation to both local and global factors. The knowledge engineer is never certain that the solution chosen is best or would "win the war." The solution is not testable in any real sense.

The EXPERT PARADOX problem consists of the fact that performers who are "merely competent" can often better verbalize the reasons for their decisions than those performers who are acknowledged as true experts. Competent performers also show generally less irritation than do experts when asked questions by a knowledge engineer. According to Waterman: "The more competent domain experts become, the less able they are to describe the knowledge they use to solve problems!" (Waterman, 1986, p. 154). Waterman referred to this phenomenon as "the knowledge engineering paradox" and said that the domain expert "... makes complex judgments rapidly, without laboriously reexamining and restating each step in his reasoning process. The pieces of basic knowledge are assumed and are combined so quickly that it is difficult for him to describe the process" (Waterman, 1986, p. 153). Similarly, Johnson (1983) reported that "... paradoxically, an increase in expertise seems to result in a decline in ability to express knowledge." Johnson called it "the paradox of expertise." Along these lines, Fraser pointed out that "knowledge acquirers may provoke resentment by rejecting the expert's description of his reasoning and pressing him to justify every conclusion" (Fraser, 1987, p. 2).

We had access to several experts who differed as to both their levels of expertise and their verbal skills. It appears that experts with higher levels of expertise operate in a nonverbal, intuitive mode. When they are queried as to the reasoning that led them to a particular decision, they are basically forced to invent a plausible story. Doing so requires them to revert to an effortful analytical mode that they have long ago left behind. They appear to feel insecure in this mode. Insecurity is aggravated when the knowledge engineer attempts to probe inconsistencies in successive stories. As a result, the expert frequently gets irritated or avoids direct answers by digressing into more or less pertinent anecdotes.

In contrast, performers at lower levels of expertise appear to operate in an analytical and more effortful mode. They seem to have traces of their reasoning readily available and, given good verbal abilities, produce without hesitation fairly detailed accounts of their decision processes. They exhibit less of a tendency to become irritated or to digress.

The knowledge engineer must be especially concerned with the CATCH 22 problem. A knowledge engineer is caught in a "Catch 22 situation" in that he cannot elicit deep knowledge unless he asks the right questions, but he cannot ask the right questions unless he has some deep knowledge. Fraser, referring to this as a "chicken and egg dilemma," pointed out, "It is difficult for the knowledge acquirer to

achieve the competence required to elicit, and meaningfully interpret, the knowledge that experts convey" (Fraser, 1987, p. 6).

In our case, the problem manifested itself in the form of increasing redundancy in expert responses. After the first two focused interviews, the subsequent structured interviews produced less new and deeper knowledge than was expected. Because the structured interviews were based on specific hypothetical decision cases, it was expected that the experts would produce not only specific decision responses but also specific, detailed, and deep reasoning explaining their decisions. These hoped-for responses were not forthcoming. Instead, much of what was learned during the last two interviews was redundant with knowledge that had already been acquired during the first two interviews.

The ACCESS problem refers to the fact that domain experts, being experts, are usually quite busy and often remotely located. Schraagen (1986) listed "the expert is inaccessible" as a common complaint of knowledge engineers. Waterman said: "Pick a nearby expert, preferably in the same city. Otherwise, consider relocating the expert for the duration of the project" (Waterman, 1986, p. 193). Such a solution, however, is often not practical even if feasible.

This was true in our case. Our situation was aggravated by the geographical distance between the knowledge engineers, who lived and worked in Southern California, and the experts, who were stationed in Texas. Commitments on either side, as well as travel restrictions, led to limited access to the experts.

The QUID-PRO-QUO problem is related to the ACCESS problem. Knowledge acquisition activities take experts away from their primary jobs. Any unit that makes experts available therefore "donates" often substantial amounts of the time of their best people and, because knowledge engineering is hard work, these SMEs are not necessarily having a good time at it. As a result, knowledge engineers may face decreasing motivation on the part of the SMEs or the SMEs' unit to support a project whose eventual benefits are difficult to visualize. In addition, as Schraagen stated, "Experts may be afraid of losing their jobs, of being replaced by computers, or they may be skeptical about the value of using artificial intelligence and computers" (Schraagen, 1986, p. 56). The solution to the problem is a quid-pro-quo which is truly useful to the experts and/or their unit. "Give the expert something useful on the way to building a large system" (Buchanan, et al., 1984, p. 165). To be perceived as "useful," a benefit must be immediate or near term, so that personnel who are currently at the unit will be able to make use of it before they get transferred. Without some type of near-term and practical quid-pro-quo arrangement, it is unlikely that a productive relationship can be maintained throughout the project.

Computer-Aided Knowledge Acquisition

Concept. The results of the unaided knowledge acquisition effort pointed to a need to improve the effectiveness and efficiency of further knowledge acquisition efforts for the project.

The UNCERTAINTY problem arises from inherent characteristics of the domain and can basically be addressed only by abandoning the idea that every decision problem has to have a solution that is in some sense optimal. The EXPERT PARADOX problem is easily overcome if SMEs are experienced enough to be competent, but not so experienced that they have already made the transition to fast, parallel, and intuitive processing. That leaves the CATCH-22, the ACCESS, and the QUID-PRO-QUO problems as targets for a different type of knowledge acquisition methodology. To solve these problems, the following functional requirements had to be satisfied:

1. A convenient scenario generator mechanism must be available which enables SMEs to employ their domain knowledge to construct the kind of rich and realistic decision scenarios which the knowledge engineers could not construct due to their lack of sufficiently deep knowledge.

2. The scenario generator must be capable of presenting SME-constructed decision scenarios to experts (other SMEs), permit them to perform the same information search processes they use in reality prior to making a decision, accept the experts' decision input, and provide a means to record the experts' reasoning.

3. The scenario generator must reside on hardware available to the experts and should not require the presence of an analyst/knowledge engineer for scenario construction or for scenario presentation.

The first requirement addresses the CATCH 22 problem by employing the expert's deep knowledge to create the knowledge elicitation stimuli. The second requirement generates a system that can function as a knowledge acquisition device and/or as a training device. By providing for a training device, the requirement addresses the QUID-PRO-QUO problem. Together with the third requirement, it also addresses the ACCESS problem, by making a convenient system available at the expert's unit that would enable a knowledge elicitation process without requiring the presence of a knowledge engineer.

The purpose of knowledge elicitation was to acquire the instructional content for the training system prototypes. The scenario generator system therefore had to have the capability to capture the decision-making process in explicit form. To get a complete picture of the expert's cognitive processes during decision-making, it was necessary to capture the expert's decision output and his underlying reasoning as well as a trace of his overt, observable activities in arriving at a decision. The system environment there-

fore had to permit the expert to engage in a pattern of interaction with the system that would, in all functional respects, be identical to the real pattern of interaction between the expert and the operational environment. For example, if the expert during actual operational decision-making would have to query specific sources of information, he should be able to make the same kinds of queries in the desired scenario generator system and receive the same kinds of information from the system as he would from actual information sources. The system environment therefore had to be designed such that it would be at least functionally isomorphic to the operational environment.

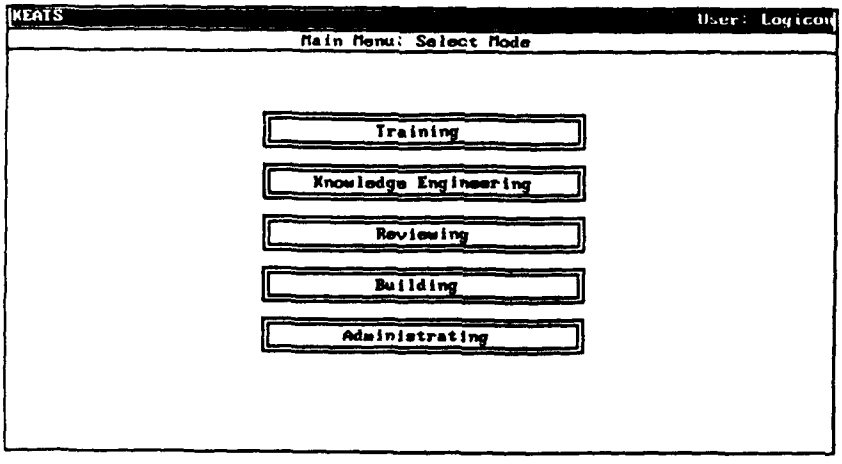
Functional fidelity would, of course, also benefit the intended use of the system as a training device. The question was whether training applications of the system would require not only functional fidelity but also some degree of physical fidelity, such as precise replication of paper forms, status board layouts, and message formats. The answer to this question is basically dependent on the training objectives to be supported by the system. If the system is to support the acquisition of procedural skills, then physical fidelity is indeed required. If the system is to support the acquisition of decision-making skills, then functional fidelity is likely to be sufficient. If the system is to support both types of objectives, then both functional and physical fidelity should be built into the system. As it appeared desirable to build a system which would support a variety of training objectives, it was decided to replicate physical aspects of the operational environment to the extent it was possible to do so without detracting from the primary task of developing a functional scenario generator system for knowledge elicitation purposes.

Implementation: KEATS. The functional requirements delineated above formed the basis for a system design which was implemented in Smalltalk V 286 on a ZENITH 248 microcomputer. Details on the hardware and software requirements for KEATS are shown in Table 2. Because this system supports both knowledge engineering and training activities, it is called the Knowledge Engineering and Training System (KEATS). The operating modes and design features of the system are described below.

Operating Modes. KEATS has five operating modes which are accessed through the system's main menu (see Figure 2). Two of the modes are designed to facilitate knowledge acquisition in the study domain: the BUILDING mode and the KNOWLEDGE ENGINEERING mode. Besides these two modes, KEATS has a TRAINING, a REVIEWING, and an ADMINISTRATING mode. The TRAINING mode is identical to the knowledge engineering mode but does not include the automatic knowledge elicitation queries. The REVIEWING mode allows on-line review and critique of transcripts collected during knowledge engineering sessions. The ADMINISTRATING mode is used to keep track of exercises, transcripts, and system users. The building, knowledge engineering, and training modes are discussed in detail below.

Table 2. KEATS Hardware and Software Requirements

SOFTWARE	KEATS APPLICATION	0.8 MB IMAGE
	TYPICAL EXERCISE	0.2 MB
	SMALLTALK/V286	0.5MB
	SMALLTALK/V GOODIES	
HARDWARE	IBM PC/AT COMPATIBLE, 286 OR 386, 3 MB RAM	
	HARD DISK, 5 1/4" FLOPPY	
	EGA, MOUSE SYSTEM COMPATIBLE MOUSE	
	IBM GRAPHICS PRINTER OR HP LASTER JET PLUS	
	DOS 3.2 OR 3.3	



Mode Selection Screen

Figure 2. KEATS Main Menu.

BUILDING Mode: This mode is available in GUIDED and UNGUIDED submodes. The GUIDED submode is intended for a novice user. It is less elegant and less efficient to use than the UNGUIDED mode, but it is easier to learn and protects the user from errors. In either BUILDING submode, the user constructs an ASOC exercise which typically covers the daylight hours of one day. Each exercise consists of two parts: a tactical situation and a script of events (see Figure 3).

Events present decision problems. The tactical situation provides the context within which the decision problems occur. Tactical situations and scripts are developed with two separate editors. Both editors work along the same principle: The user selects objects from a collection of standard objects and customizes them for the exercise. For example, as a part of defining the tactical situation, the user may select a standard armor division which he customizes by giving it a name, a location, and a current combat strength; as a part of defining the script, the user may choose an air request event which is instantiated by a specific origin, a target, and a desired time-over-target.

The definition of the tactical situation must contain a minimum set of mandatory scenario objects in order for an exercise to run in the training or knowledge engineering modes. A special exercise validation routine is available which ensures that the minimum set is present.

The result of exercise building is thus a chain of events which occurs in a specified tactical environment. Any tactical situation can be combined with any number of different scripts. Any defined tactical situation and script can be edited. Once a small number of different exercises exist, additional exercises can be built very efficiently by modifying the existing ones.

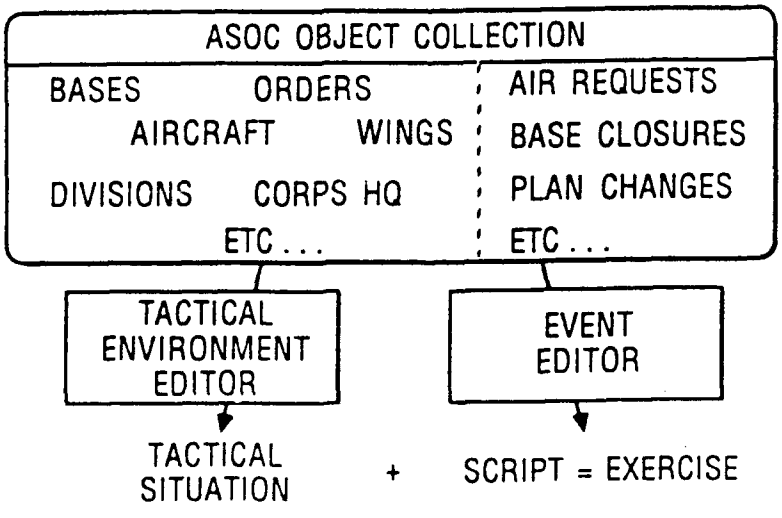


Figure 3. KEATS Building Mode.

KNOWLEDGE ENGINEERING Mode: In this mode, the user "runs" a previously built exercise. Exercises unfold in two phases: an initial orientation phase and a subsequent operations phase. The general flow of an exercise is shown in Figure 4.

During the *Orientation Phase* of an exercise, the user first familiarizes himself with the overall tactical situation. This initial step simulates the beginning of the day shift in the real world. This familiarization process is enabled by making available the same information resources that are present in the real world; i.e., by supplying briefings and orders to review, by enabling question-and-answer interactions with members of the ASOC team or with external agencies, and by reviewing status boards and maps (maps are currently supplied on paper).

The result of the orientation phase is an initial *situation assessment* which, in turn, leads to an initial *plan of action*. The plan of action usually entails preparing some of the air resources for immediate tasking by putting them on various levels of alert.

The user can take as much time as he wants for this phase. Before he can begin the second phase of the exercise, the knowledge engineering mechanism elicits his initial situation assessment, his plan of action and the specific conditions that led him to put certain airplanes on certain alert states. The user responds by typing the requested information into forms with a word-processor-like facility.

During the *Operations Phase* of the exercise, the user reacts to events as he triggers them. Events show up as messages which arrive at a *message desk*. He reacts to events as he would in the real world by accessing various information resources to confirm the event, to gather additional information about it,

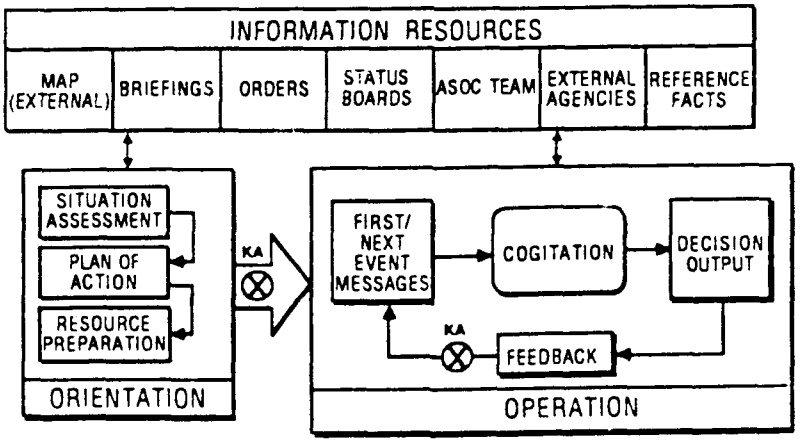


Figure 4. Exercise Flow in KEATS.

and to determine solution options. After this period of "cogitation," he enters a decision response to the event by telling some other agency to do something; i.e., by issuing a specific order.

KEATS evaluates the user's response by checking for violation of physical constraints. For example, the user may respond to a request for air support by tasking some wing to fly a mission with four airplanes of a specific type that must be at their target during a certain timeframe. KEATS checks whether these airplanes are indeed available at that wing and whether they can make the desired time-over-target (TOT). The results of constraint checking are returned to the user as feedback messages. If the tasked mission does not violate any physical constraints, the user receives a message which indicates that the wing accepts the tasking. Otherwise, the wing sends a refusal message to the user.

After the user has reacted to an event, he is queried by the knowledge engineering mechanism -- first, with respect to the specific conditions that led him to the decision he made, and second, with respect to any changes in the situation assessment or plan of action that might have been triggered by the event. Again, he types in his responses. All user-system interactions and all user inputs are saved on a transcript which is, in fact, a detailed decision-making protocol.

The knowledge engineering queries in each exercise are designed to elicit knowledge from two hierarchically related levels of decision-making shown in Table 3. On the macro level, the ASOC battle manager deals with the "big picture" by assessing (and constantly re-assessing) the overall tactical situation for the ASOC and by deciding on (and changing) a plan of action which is consistent with this

Table 3. Knowledge Elicitation Queries in KEATS

KNOWLEDGE LEVEL	QUERIES DURING	
	ORIENTATION PHASE	OPERATION PHASE
MACRO LEVEL ("BIG PICTURE")	WRITE INITIAL SITUATION ASSESSMENT WRITE INITIAL PLAN OF ACTION	WRITE CHANGES TO SIT. ASSESSMENT WRITE CHANGES TO PLAN OF ACTION } AFTER EACH DECISION OUTPUT
MICRO LEVEL (SPECIFIC EVENT)	RESOURCE PREPARATION: (PUTTING AIRCRAFT ON ALERT) STATE CONDITIONS SHOW BOUNDARIES	IF NOT ROUTINE: STATE CONDITIONS SHOW BOUNDARIES IF ROUTINE: GIVE OPTION COMPARE } AFTER EACH DECISION OUTPUT

assessment. On the micro level, the battle manager decides on what to do about a specific event. Decisions on this level are guided and constrained by decisions made on the planning or macro level.

This knowledge acquisition strategy is essentially based on the two modes of decision-making (i.e., planning and operations) that are found throughout the TC² System. The battle manager must form and continually update a global perception of the tactical situation and must formulate a plan for dealing with that situation. He must solve any particular decision problem by considering the local, short-term parameters of the problem within the context of the "big picture"; i.e., his overall situation assessment and plan.

It follows then that knowledge acquisition must aim to capture both of these aspects of expert knowledge. It must capture the expert's global or macro perception of the tactical situation and the general heuristics for dealing with the overall situation. It must also capture the expert's local or micro perception of the specific decision problem (presented by a given event) and the heuristics that apply to solving decision problems at that level. Finally, knowledge acquisition must capture the interaction between the context and the specific problem; i.e., the reasoning that links the macro and micro levels.

The knowledge acquisition procedures currently implemented in KEATS are designed to capture these knowledge elements. KEATS queries the subject first on his overall perception of the situation, by asking him for his ASSESSMENT of the situation; it elicits general heuristics by asking for his PLAN OF ACTION. As subsequent events occur, the subject is asked to provide any changes or updates he might want to make to his assessment or plan of action. To elicit knowledge on the micro level, the subject is queried after each decision for the specific CONDITIONS that have led to the decision. It is expected that this query will yield responses on both levels, micro and macro, and that it will provide insight as to how these two levels interact. This interaction is further explored with the final type of query, where the subject is asked to indicate what conditions would have to change (and how much) to make his decision invalid (i.e., to trigger a different decision).

Events may not always require a decision but rather, a routine or standard response. Those are typically the events that fit perfectly with the current situation assessment and plan of action. They are, in other words, expected and thus merely trigger a planned response without requiring a choice between response options. If this occurs during a knowledge engineering session, the subject can so indicate. As a result, the subject does not have to repeat a routine set of conditions to justify and explain his response. Instead, the subject is asked to provide an alternative to the routine response and compare the two. This type of query is designed to elicit additional micro-level reasoning. It also prevents the subject from

answering the knowledge engineering queries in routine cases by merely referring to answers to some earlier query.

Evaluation

KEATS was evaluated as a knowledge acquisition support system on two separate occasions, with different methods and evaluation objectives. Both evaluations are described below.

First Evaluation. Two ASOC experts used the system for 2 consecutive days at the contractor's facility. Each expert was independently (in separate rooms) subjected to two different exercises that had been developed by the contractor, rather than by the experts themselves. On Day 1, Expert A ran Exercise 1; Expert B ran Exercise 2. On Day 2, the experts switched exercises so that both experts did both exercises on different days. Each expert was assisted during both days by a contractor analyst/ knowledge engineer. The analysts were there to observe the experts and to provide "over-the-shoulder" assistance in system use if required. The analysts also taped any discussions with the expert and/or the expert's monologues (the experts were encouraged to "think out loud"). The tapes were made to determine whether significant information was getting lost during the relatively laborious keyboard input of reasoning.

Knowledge Product. Overall the two experts produced 29 pages of transcript. Expert A produced approximately seven more transcript pages than did Expert B. Expert A also processed five more events than Expert B processed during the same time (Table 4). Expert A had had a prior exposure of several days to an earlier version of KEATS.

Table 4. Productivity During Knowledge Elicitation

Output	Expert A	Expert B
Events processed	11 + 11 = 22	10 + 7 = 17*
Number of pages of transcipts	9 + 9 = 18	6 + 5 = 11

* Exercise 1 + Exercise 2 = Total

Macro-Level Knowledge: During the Orientation Phase of the exercises, the two experts produced a total of four Initial Situation Assessments (ISAs) and four Plans of Action (POAs). ISA and POA length was about six lines on the average; i.e., the length of an average paragraph of text. Length differences between ISAs and POAs or between experts were negligible.

Analysis of the assessments and plans revealed common content elements across exercises and experts. Assessment consisted basically of three parts: statements defining the probable demand for air support, statements evaluating the anticipated supply of air resources, and identification of replanning triggers or factors that should be watched because they might trigger a revision of whatever plans were made (see Table 5).

Table 5. Content Elements in Situation Assessments

Element	Expert A		Expert B	
	Exercise		Exercise	
	1	2	1	2
Demand asssessment				
ID of need[a]	X[b]	X	X	X
When		X	X	X
Division	X	X	X	X
Priority	X[c]	X	X	X
Supply assessment				
Type of a/c	X[c]	X[d]	X	X[e]
Ordnance	X[c]	X		X
Planned allocation to meet demand	X		X	X
Time to target area			X	X
Replanning triggers (e.g., Wx at 1100, end of attacking period)				
What	X	X		X[f]
When	X	X		X[f]

[a] role, i.e., offensive/defensive attacking/holding is an important factor.
[b] did not note attack time and looked at strength and fire priority.
[c] stated as action conditions.
[d] "mix."
[e] ID of distringuishing features.
[f] on notepad.

Plans consisted basically of a partial mapping of air resources by squadrons to front-line divisions. The two POAs (by each of the two experts) for one of the exercises are shown in graphic form in Figure 5. They represent clearly very different ways of dealing with the same situation, but they are constructed of the same elements. The ISAs which were generated prior to the plans were also different but could not

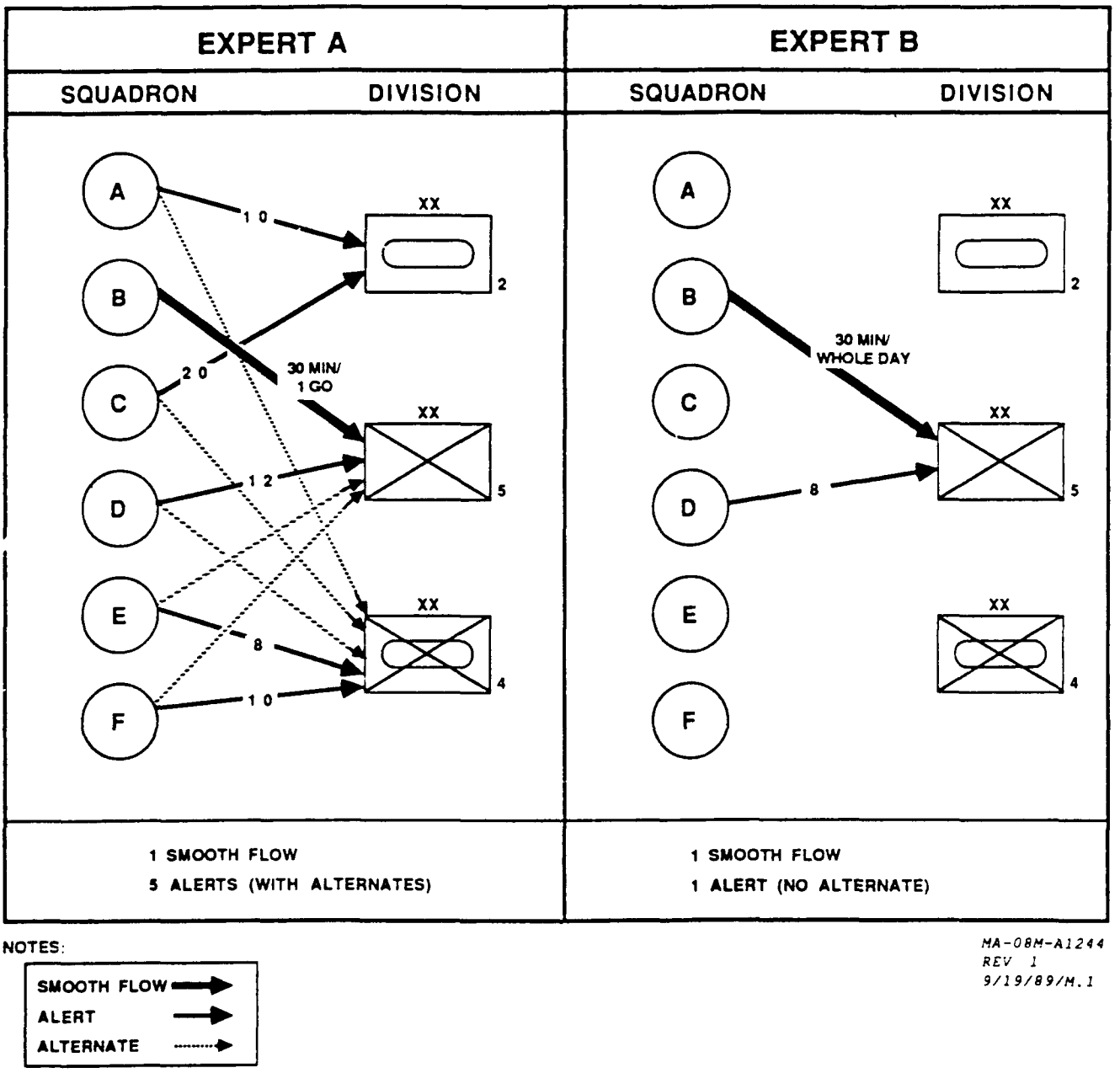


Figure 5. Plan of Action Differences in Identical Situations.

logically account for all plan differences. It was inferred that other macro knowledge not tapped by the current elicitation strategy in KEATS can account for unexplained plan differences.

During the Operations Phase of Exercise 1, only one change in SA and POA was noted (by Expert A). In Exercise 2, both experts changed SA and POA twice but at different times in the exercise (i.e., in

response to different stimuli). The changes always involved a revision of the initial demand assessments and resulted in a revised plan for allocation of air resources.

Assessment and planning thinking on tne macro level was also found in some of the responses to queries on the micro level. One of the experts used the Notepad facility extensively and noted some of his planning and assessment thoughts there.

Tape transcripts and KEATS protocols contained the same relevant information. The tape transcripts also contained discourse unrelated to SA and POA.

Micro-Level Knowledge: Micro-level knowledge was elicited for expert decisions in response to air requests. Experts were asked to list and rank-order the conditions that prompted their decisions. Altogether, the experts made 28 tasking decisions and justified them by listing 135 conditions.

There were noticeable differences between the two experts. Expert A responded with an average of about four conditions, verbalized (thought out loud) his reasoning before entering the decision, and then entered his reasoning. Expert B responded with an average of about six conditions, made and entered his decision with a minimal verbalization, and then rationalized his decision post facto.

No differences between tape transcripts and KEATS protocols were noted, except for the presence of discourse unrelated to the decision tasks in the tape transcripts.

Subsequent analysis of the tasking decisions and conditions produced 64 rules on nine different topics related to decision-making in response to air requests (see Table 6).

Table 6. Number of Rules Obtained by Topic

Order of tasking	4
Requests before tasking	6
Putting aircraft on alert	2
Tasking beyond NLT	2
Selection of assets	26
Amount of support	11
Request refusals	3
Smooth flow	5
Diverts	5
Total	64

NLT = No later than time

Additional Knowledge Products: The first trial of KEATS as a knowledge engineering support tool was also a formative evaluation of the KEATS system. Two types of information were collected:

1. Information relating to the fidelity of built-in functions and facilities within the KEATS environment, and

2. Information relating to the fidelity of exercises that were presented to the experts (and constructed by means of the KEATS building function).

Both types of information can be viewed as knowledge engineering results in their own right. The trial produced a total of 23 items indicating changes to be made to KEATS and 15 items indicating changes to be made in existing and future exercises.

Cost. The cost of acquiring the knowledge product is expressed in manhours spent during the trial (i.e., during knowledge elicitation) and manhours spent after the trial (i.e., during knowledge analysis and formulation). These costs are illustrated in Table 7.

Table 7. Manhour Cost of Knowledge Acquisitions

	Experts	Analysts (Knowledge Engineers)	All
Knowledge Acquisition	29.4	29.4	
Analysis and Formulation		102.0	
Totals	29.4	131.4	160.8

Second Evaluation. During the second evaluation, an ASOC expert (Expert A of the first evaluation) spent about 6 hours modifying an exercise. Developing a new exercise is a lengthy process involving extensive planning. The expert was already familiar with the selected exercise through his previous use of the KEATS knowledge engineering mode. He chose to revise the contractor-developed exercise to keep within the scheduled 1-day evaluation time limit.

During the first 3 hours, the ASOC expert primarily changed, extended, and added to the fact and tactical environment elements in the exercise. Values were modified for accuracy, a new aircraft type was added, and three new tactical elements were defined. The remaining time was used to script and insert eight new scenario events. Seven of the events developed an emergent tactical situation (an air insertion of enemy troops) that would require decision-making during a period of resource shortages.

During the same time, two ASOC officers were asked to fill out a questionnaire after using KEATS for about 4 hours. The questionnaire consisted of statements about the completeness, fidelity, and usefulness of KEATS. Ratings on a 5-point agree/disagree spectrum, along with explanations and suggestions, were solicited.

Some explanations as to how the written evaluations are used here is warranted: ASOC member #1, Expert A of the first KEATS evaluation, has had 8 years of experience with the 712th ASOC, has participated in numerous exercises, and knows as much about what the ASOC does as anyone who has not actually been in combat with an ASOC. He is the prototypical "expert." ASOC member #2 has had minimal prior exposure to KEATS, has been assigned to the ASOC for about 1 year, has limited exercise experience, and has never worked the day shift (when tasking decisions are made) during an exercise. He knows what the ASOC does but not how to do it himself. He is the prototypical "novice." Because of this lack of ASOC experience, ASOC member #2's evaluation of the knowledge engineering in KEATS is not relevant. He has little knowledge about ASOC decision-making for close air support and hence cannot tell us whether KEATS acquired that knowledge or not. His opinions concerning the usefulness of the exercise Build function lack credibility because he is not experienced enough to know how to build an exercise. He can, however, add valuable input in evaluating the training function since this is the area in which he would find KEATS of use to him. In the discussion below, then, the references to expert evaluation of knowledge engineering and exercise building are those of the expert, ASOC member #1.

Knowledge Engineering with KEATS. The evaluation of KEATS as a knowledge engineering tool rather than a knowledge acquisition device is intentional and implies a broader scope than strictly acquisition. Knowledge engineering encompasses the areas of knowledge acquisition, knowledge representation, and using the knowledge to construct an intelligent agent. Although the immediate goal of KEATS is knowledge acquisition (or more specifically, knowledge elicitation from ASOC experts), the ultimate goal is to use that knowledge in a training system. Using the knowledge requires having a consistent and complete representation of the knowledge. If KEATS yielded knowledge that was difficult to represent and impossible to use in a training system, it would be of little value.

Correctness of the Knowledge Obtained by KEATS. Correctness of the knowledge that the experts impart is critical to the later use of that knowledge in a training system. If the KEATS environment distorts what the expert did and why he did it, such that it differs from what he would do in a real situation, then a training system based on this knowledge will be inherently deficient. Evaluation of correctness can come only from the experts themselves. The two ASOC members had two sources from which to evaluate correctness -- the KEATS transcript in which they had directly entered decisions and given reasons for

those decisions, and the rules that were derived from the transcripts (e.g., Appendix A). The expert (ASOC member #1) rated the correctness of his decisions in KEATS as excellent compared with those in a real-life situation. The correctness of the reasons given for the decision received the middle rating (3 of 5) because he said that needed information about the situation was not always available and hence required a "best guess." However, he said that there was no inherent limitation in the technology that would limit correctness. Further refinement of the KEATS environment to provide currently missing information should alleviate the best guess problem.

Completeness of the Knowledge Obtained by KEATS. It is important to know whether KEATS would allow the acquisition of all the knowledge the experts use in decision-making. If it does not, then some additional means of knowledge acquisition must be employed to fill in those gaps. This is an area, again, that only an expert can judge. Because of the small number (2) of exercises that the experts ran, the fact that the exercises were built by Logicon personnel rather than by experts, and the fact that the experts had time to go only partway through those exercises, it would be surprising if anything like complete knowledge of ASOC decision-making was obtained. In fact, the expert rated KEATS as medium (3 of 5) for having obtained "all" the decision-making knowledge. The expert did not give specific areas that were missed but suggested checking the major events in a large CPX. He agreed (4 of 5) that more exercises with different conditions would improve completeness. He was neutral (3 of 5) as to whether KEATS design limitations reduced completeness but felt that the implementation could be modified as experience is gained with KEATS. He felt that the technology itself (i.e., single-user, computer-based) did not limit completeness.

Depth of Knowledge Obtained by KEATS. It is important that the proper depth of knowledge is obtained so that it will be useful in a training system. If the knowledge is too shallow, the trainee may learn what to do in a specific case but not know the basis for the decision and not be able to extend his knowledge to slightly different situations. If the knowledge obtained is too deep, it quickly becomes absurd (e.g., Why did you set up a smoothflow to Division 1? Because they are attacking and hence need continuous support. Why are they attacking? So they can capture the enemy supply source. Why do they want to capture the enemy supply source? So we can win the war. Why do we ...). In an automated knowledge elicitor, arriving at the appropriate depth depends on the purpose for which the knowledge will be used and is determined by the questions asked the expert. In KEATS, the purpose was to obtain knowledge useful in a training system. The expert was asked for conditions that led to a particular tasking decision, how much those conditions had to vary to cause another decision, what were other possible tasking options, and how the options compare to the original tasking.

Evaluation of depth is from two sources: what the expert said and usefulness of the knowledge in later training system development. With respect to the expert's opinion, he strongly agreed (5 of 5) that the answers and reasons that he gave reflected what he would want a trainee to know if making the same decision. He felt that other questions would not have yielded more useful answers, although he did indicate that the meaning of the question must be well understood by the subject answering. It is noted that the exercises reported here were run with a contractor representative present to try to clarify what was desired by the KEATS questions if the experts' answers appeared to miss the point. Also, the expert thought the sequence of making a decision and immediately being questioned about it was good and necessary, though at times irritating. Use of the knowledge for training was confirmed in the SuperKEATS system. SuperKEATS is a simulation learning environment based on the knowledge gained from the KEATS system. Various rules of tasking (as given in Appendix A) were implemented in SuperKEATS to provide suggestions, and other rules were encoded to generate tasking options and give feedback to the user. Hence, the depth of knowledge was directly useful for a training application.

Efficiency of Obtaining Knowledge Using KEATS. Efficiency of obtaining knowledge is a function of both the expert's time needed to run an exercise and the knowledge engineer's time to extract and represent the data. Usually, as it was in this case, the critical factor is the expert's time. Doing expert work (e.g., in the case of the military, preparing to defend the country) is the expert's primary duty, with other work on an as-time-permits basis. Knowledge engineers are typically assigned full-time to a project so may devote whatever time is required to knowledge extraction and representation. Because the expert's time is very limited, it is important to gather as much information as possible in the time that he has available.

From the expert's point of view, KEATS was somewhat tedious to use, with the continual questioning that requires keyboard input. SMEs are not typically good typists, and typing responses during the exercise interferes with the continuity between events. Also, in each exercise the first 2 or 3 hours were spent in the assessment/plan phase where much of the information was similar between exercises. One suggestion made by the expert was to be able to start an exercise at a specific time in the day, with the situation all set so that knowledge of how to treat a specific situation can be obtained. With the current way of starting at the beginning each time, neither expert was able to get beyond an exercise time of 1000 in a full day of running. In retrospect, it might have been better to allow the experts to continue the exercise into the second day that they had available rather than starting another exercise.

In spite of these problems, when asked whether a different approach would have been better, the expert responded: "I feel that approach is good. It does not allow you to abandon a line of questioning

without at least thinking of the potential different responses. It may be irritating at times, but it is necessary." Efficiency, then, is a question of sufficiency: Is KEATS efficient enough to get the job done of acquiring complete and accurate knowledge about ASOC decision-making for close air support? Getting the job done would obviously require more than 2 days of two experts' time. It would also require a varied set of exercises to push the expert to decide on special cases and exceptions. Knowledge elicitation using KEATS may be hard and somewhat tedious work, but it gets the job done. Interviewing or other approaches to acquisition of decision-making knowledge in a complex environment may be easier work, at least for the expert, but, in our experience, are very inefficient in getting at the detailed knowledge of when and why decisions are made.

Effectiveness of KEATS in Obtaining Knowledge. From the knowledge engineer's point of view, KEATS appears to be a very effective way to collect knowledge. The transcripts provided a dense source of information both from the actions/decisions of the experts and the conditions/options given for the decisions. Using the conditions supplied by the experts, it was fairly easy to represent the tasking knowledge in terms of rules. Obtaining about 64 rules about tasking from the limited number of events (about 10) that the experts ran each exercise shows that a KEATS transcript is an excellent source of knowledge about decision-making.

An additional method for acquiring knowledge was available when the rules were sent back to the experts to review. By looking at the transcript, the experts were able to compare the rule derived by a knowledge engineer with what they did and what the situation was. Though many of their comments corrected misunderstandings of intent, they also opened up further knowledge acquisition by generalizing beyond the specific case encountered in the exercise. Grouping the rules by topic and then asking questions about seemingly overlapping rules was an appropriate and effective way to pinpoint subtle differences between rules and the intention of the expert when making the decision.

Building Exercises with KEATS. One of the design goals of KEATS is to make building an exercise easy enough so that an expert would be able to generate situations he thought to be critical to the ASOC decision-making process. However, when an ASOC expert attempted to build a KEATS exercise from scratch and use it for ASOC training, either time constraints (setting aside about a week to do only that) or motivation kept him from completing this project. Evaluation of the Build function, then, is based on 6 hours of exercise modification and results of the expert's written evaluation.

Correctness of the Objects and Events Used to Build Exercises. The objects used to build an exercise must have the characteristics necessary for ASOC decision-making. For example, if a squadron does not have an attribute identifying the number of turns that its aircraft are going to do that day, then

the number of sorties planned for the day would not be known. The ASOC expert rated KEATS highly (4 of 5) for correctness of the objects and suggested that the forward air controller (FAC) should have a role, and that capabilities, attributes and the operational status of a base should include limited operations as a value. He rated the correctness of the events as excellent (5 of 5) but also had some suggestions for improvement. It is to be expected that intensive ASOC building of exercises will bring various suggestions for modifications and improvements of which the developers were unaware.

Completeness of the Set of Objects and Events Used to Build Exercises. All the types of people, organizations, things and types of events need to be available so that relevant exercises can be created. The developers included the objects and events known to them from earlier knowledge acquisition efforts (interviews, etc.) and from a review of KEATS by the ASOC officers, but regarded KEATS development as an iterative process where part of knowledge engineering would be feedback from the experts on what else was needed in the environment. Because of the small amount of expert use of the KEATS, this feedback was minimal. The expert rated KEATS average (3 of 5) and good (4 of 5) on completeness of objects and events, respectively, and suggested that events might be expanded to include, for example, inflight reports.

Usability of the Objects and Events for Building Exercises. Did KEATS permit the objects to be used in the ways needed to build an exercise? The expert rated this capability as excellent (5 of 5), without comment. The KEATS Guided Build Mode was generally easy to use. The expert was able to access objects quickly and easily enter modifications via menu selection or keyboard input.

Modification of Objects and Events to Change an Exercise. The expert rated KEATS as excellent (5 of 5) in this capacity, without comment. KEATS allowed building of a tactical situation that the system developers had not considered. Information update events were used to communicate information about the developing situation. This points out the flexibility and extensibility that is available in KEATS.

Efficiency of Building an Exercise. The expert rated the efficiency of building an exercise with KEATS as good (4 of 5), without comment or comparing it to any other scenario generation methods with which he was familiar. Building an exercise is a very time-consuming task. First, an overall concept of what should happen in a war, throughout the day, must be conceived. Then the air units that will be available at various bases need to be laid out. Then, in KEATS, all the objects, including information items like briefings and orders must be created. Finally, the events must be created for the entire day. This is particularly time-consuming for air requests which must have locations from the map. Depending on the experience of the user in understanding Army tactics and event likelihood, the process can require anywhere from several days to several weeks. The great advantage in KEATS is the ease with which an

existing exercise can be modified to include varied situations. This was demonstrated by the expert when he modified the KEATS exercise as indicated above.

Phase III: Developing Training Methodologies

KEATS provided a means to acquire the content that should be communicated by a decision training system. The issue as to how this communication should be structured is a matter of defining an appropriate *instructional strategy*. Once such a strategy is defined, functional specifications for a delivery system can be articulated and the system can then be developed. Below, we examine the concept of instructional strategy in general, focus on the practice component which must be present in any instructional strategy, and derive a set of molecular strategy requirements which must be satisfied by a practice environment for decision-making skills. We then describe how the desired practice environment -- called SuperKEATS -- was developed. Finally, a comparative analysis illustrates the differences and similarities between SuperKEATS and KEATS (in its use as a training vehicle).

The Concept of Instructional Strategy

An instructional strategy is basically a set of rules which prescribes a particular sequence of instructional elements. Frank (1969) used the term "teaching algorithm" to describe the notion of an instructional strategy. Aagard and Braby (1976) spoke of "Learning Guidelines and Algorithms for Twelve Types of Training Objectives." They presented flowcharts depicting sequences of learner and instructional system activities which have the flavor, albeit not the rigor, of formal algorithms. Merrill (1983) favored the term "organizational strategy" and defined it as "decisions involved in the design of learning activities, including the types of displays to be presented to the student, the sequence of these displays, the topics to be included, the sequence and structure of these topics, the type of practice, the nature of feedback, and other decisions regarding the nature of the presentation."

Instructional strategies can be defined at various levels of resolution. Frank (1969) and Merrill (1983) distinguished between macro and micro strategies. Macro strategies define the sequence of subject-matter topics (Merrill, 1983) or objectives (Brecke, 1976). Micro strategies define the sequence and form of instructional events within an instructional module for a topic or objective (Brecke, 1976). Other authors have also spoken of *molar* and *molecular* levels of instructional strategy (e.g., Gropper, 1974). In essence, instructional strategies can be defined on a continuum of granularity: on the level of a curriculum as a sequence of courses; on the level of a syllabus as a sequence of lessons; on the level of a lesson as a sequence of objectives; and, finally, on the level of an objective as a sequence of instructional activities or

elements. In the present report, the terms molar and molecular will be employed to avoid confusion with the macro and micro aspects of decision-making at the ASOC.

On the molecular level, instructional strategy follows a basic and universal paradigm of three successive phases: Presentation, Practice and Testing (Merrill, 1983). Variations in instructional strategies occur one level below this paradigm within each of the phases. In Phase III (of the present project), we focused on the molecular level of instruction and on that level on the *practice* phase. As indicated before, our goal was to design a suitable practice environment for decision-making skills. We were, therefore, concerned with the relatively narrow problem of defining molecular instructional strategies that would optimize learning of decision-making skills during the practice phase of instruction.

Instructional Strategy Based on Component Display Theory

Prescriptions for instructional strategies can be obtained from two sources: Instructional Design Theories and Task Analyses. Instructional design theories and prescriptive theories are generally founded on two types of descriptive theories: theories of learning or skill acquisition, and theories of skill performance.[2] The strategy prescriptions which can be derived from instructional design theories generally address molar as well as molecular levels.

The most detailed instructional design theory that is currently available for the molecular level is Merrill's Component Display Theory or CDT (Merrill, 1983). CDT is based on Gagne's postulate (Gagne & Briggs, 1979) that different types of learner outcomes require different conditions of learning and on a classification scheme for learner outcomes that is known as the Performance-Content matrix (Figure 6). Each cell of the matrix defines a different class of learner outcome or type of training objective. CDT offers instructional strategy prescriptions for each type of objective, where the prescriptions are formulated in a "language" that includes four *primary presentation forms* and eight *secondary presentation forms*.

Training objectives aimed at the acquisition of decision-making skills can be classified in terms of Merrill's Performance-Content matrix; therefore, a corresponding molecular strategy for decision-making objectives should ipso facto be available. The most appropriate category for decision-making objectives is the USE-PRINCIPLE cell.[3] For the performance category USE, CDT prescribes (regardless of content

[2]For an interesting discussion of the differences between descriptive and prescriptive theories of learning and instruction, see Reigeluth, 1983, pp. 21-25.

[3]Use means to use a general rule to process specific information" (Merrill, 1983, p.303). "Principles are those cause-and-effect or correlational relationships that are used to interpret events or circumstances" (Ibid., p.288).

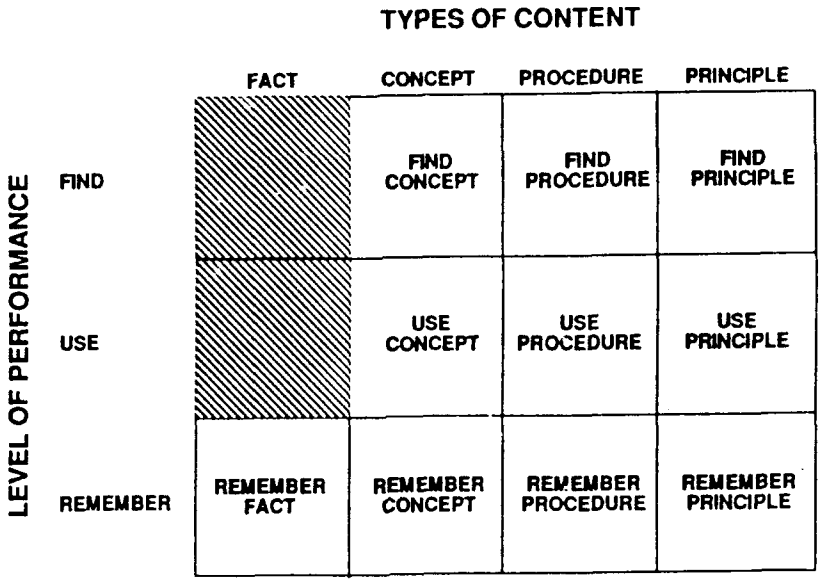


Figure 6. Performance-Content Matrix

category) Iegs.N for the practice phase; i.e., the use of a set of two or more "Inquisitory" examples *(egs)* or instances. The specific form of these practice items is prescribed as follows:

> The instance practice for a principle entails the presentation of a problem situation to the student. The name of the principle may or may not be given, and the student is asked to explain what happens. This explanation may take the form of a prediction about the outcome, or it may take the form of a solution to the problem. Representation [again] plays an important role. If it is impossible to have the actual objects and events present, then they must be represented. The representation must be sufficient so that all of the critical aspects of the cause-and-effect relationship are represented to the student. (Merrill, 1983, p.318)

This is an admirably detailed prescription if one considers the myriad of different learning outcomes that might fall under the USE-PRINCIPLE class. It is, however, far too imprecise to serve as a principled design specification for a particular subclass of skills such as decision-making skills.

Cognitive Task Analysis

At this point, the second source of instructional strategy prescriptions, Task Analysis, becomes useful. Decision-making tasks require a particular type of task analysis. The conventional methods of successive task decomposition into smalier and smaller steps (resulting in hierarchical and/or flowchart representations) have been found to be ineffective for complex, cognitive (and thus covert) tasks. These types of tasks require a method that elucidates the essential cognitive events that occur during task performance. Lesgold et al. (1986) called this type of analysis "Cognitive Task Analysis" (see also Means & Gott, 1988).

A cognitive task analysis of the targeted decision-making skill emerged as a byproduct of the knowledge acquisition efforts during Phase II of the present project. The intent of the knowledge acquisition phase was to obtain, in explicit form, expert knowledge employed during task performance. Such knowl-

edge was obtained and, in the course of its acquisition, a model of the cognitive events involved in the target task emerged. Thus, there was neither the intent to perform a cognitive task analysis nor a desire to follow Lesgold's methods; yet, the initial interviews naturally focused on the technical and cognitive aspects of the target decision domain and the specific decision task. The product of these efforts was a detailed understanding of the target domain and a cognitive model of the target decision task. This product is briefly summarized below.

The Target Training Objective. In order to keep the effort within manageable limits, the prototype system had to be aimed at a single and specific target training objective. This objective had to be representative of the decision skills exercised on the offensive side of TC^2 in the Air Force. In earlier work, task analyses were performed for a number of positions at the Allied Tactical Operations Center (ATOC). These analyses and the work performed in Phase II of this project (Brecke et al., 1989) showed that the ASOC FDO performs a task that is nearly identical to several tasks occurring in the ATOC. This task was used as a basis for specifying the following specific training objective as the target objective for the training system prototype:

> *Decide on the disposition of air requests over an entire day operations shift, given a simulated tactical situation for an ASOC and the Army Corps it supports.*

This objective is valid not only for the FDO but for all officers in the ASOC van. It represents the essence of what the ASOC does and it is an activity to which all must contribute in a coordinated fashion. The ASOC van team must, in fact, have a shared mental model on how to do this task. The objective is therefore generic as far as the ASOC is concerned. The objective also generalizes easily to decision tasks performed in the ATOC.

The objective does not state a performance standard. The measurement and evaluation of decision-making performance was left open as a research issue that should be investigated with the prototype training system.

Cognitive Model of the Target Task. The target task represented by the objective above is summarized in Figure 7 and the description that follows.

The FDO's decision-making task is initiated by *Eliciting Stimuli* which consist of messages passed to the ASOC via a number of communication links. These messages consist of air requests (i.e., requests for air support from units of the Army Corps supported by the ASOC) and other messages which inform the ASOC of changes in the availability of air assets. The messages also include feedback on the results of previously tasked missions. We focus here on air request messages.

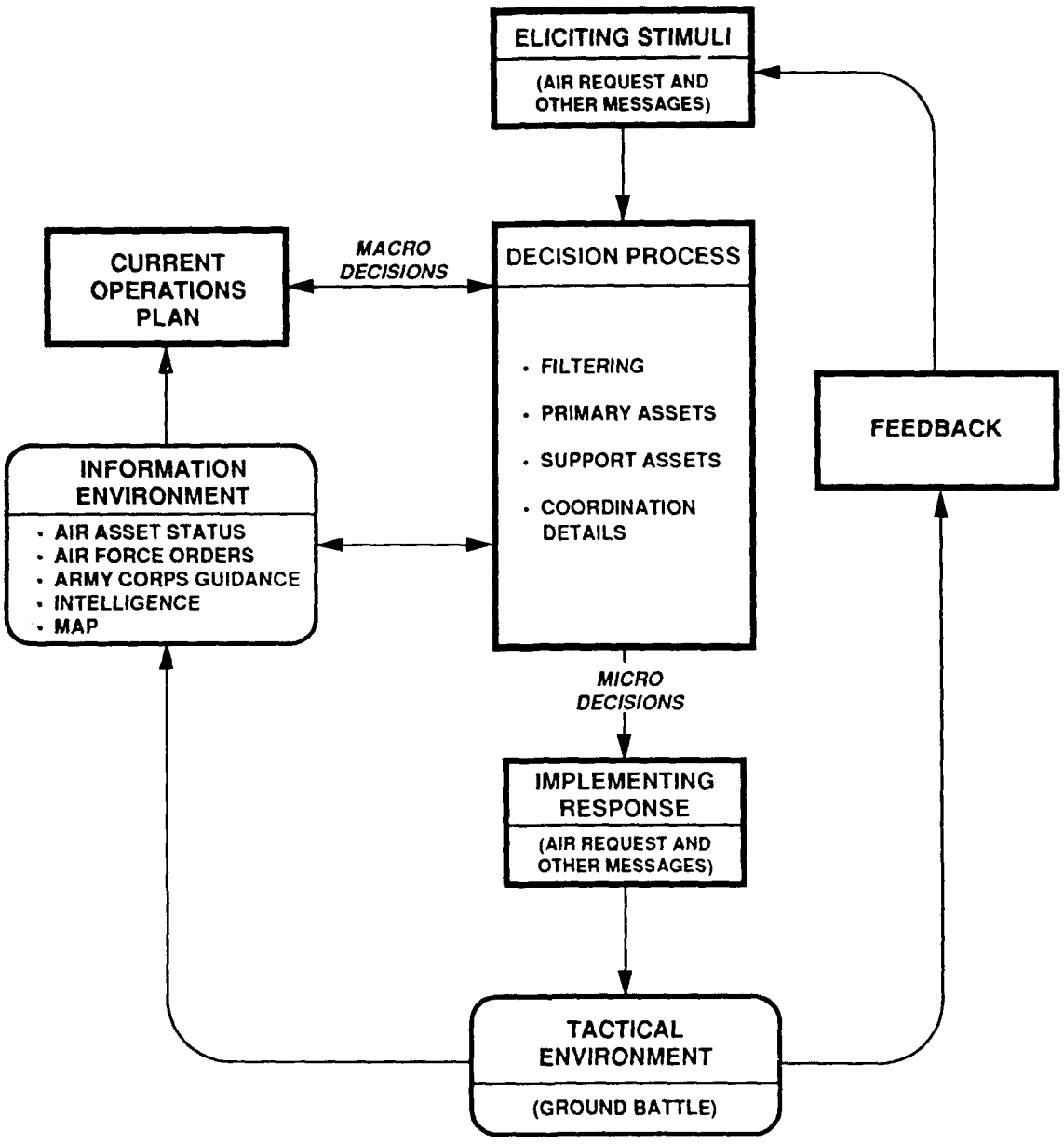


Figure 7. Cognitive Model of the Target Decision Task.

These messages elicit a *Decision Process* which is roughly subdivided into four stages. During the first stage, a verification and priority filtering process takes place in which other members of the ASOC team participate. The request must be verified as legitimate and its priority in terms of time and competing requests must be determined. The request may at this point be refused or further clarification may be requested. If the filter is passed successfully, the FDO locates and selects suitable *primary assets* (airplanes). He then determines whether *support assets* (e.g., electronic warfare and other defense suppression assets from either the Army or the Air Force) are required and available. Finally, *coordination details* regarding timing, contact points, and radio frequencies are determined. The request may get refused anywhere along this process, depending on asset availability and mission priority.

During this entire process, the FDO must have access to an information environment which must contain at least the following major types of information: Air Asset Status (available from status boards and directly from Wing Operation Centers or WOCs); Air Force Orders (Air Tasking Order or ATO, Daily Operations Order or DOO); Army Corps Guidance (plans and fire priorities); Intelligence (on enemy plans and movements); and, finally, an image of the tactical situation represented by means of maps.

Also, along the way through this process the FDO must be aware of the current *operations plan*. His decisions regarding particular air requests (i.e., his decisions on the micro level) should be guided by the current overall plan, which is a reflection of macro decisions. He must constantly reassess this plan for its continued viability in the face of new incoming information and decide whether it is time to change the plan and how it should be changed. This *macro* decision process goes on in the background while the FDO deals with the micro decisions required to satisfy particular requests.

The output of the decision process is an *Implementing Response* which consists of various types of air tasking messages that are sent via communication nets to the operational units (WOCs or squadrons) that own the airplanes which will fly the air support missions.

The implementing response thus triggers an operational response which has some effect on the *Tactical Environment* (i.e., on the course of the ground battle). If airplanes are lost during missions, the air asset status is affected as well.

Feedback on the effects of the operational responses reaches the FDO by way of mission reports which are issued either while missions are still airborne or after their return to their base of origin or to an alternate base. Further feedback reaches the FDO via intelligence reports and via changing plans and fire priorities from the Army Corps.

A particularly important source of feedback is the changing pattern of subsequent air requests. For example, an area may previously have produced very frequent requests on targets that seemed to be moving more and more into friendly territory. The FDO has responded to these demands with high levels of air support. If, as a consequence, the frequency of requests decreases and if the targets appear to be moving out of friendly and into hostile territory, then the FDO can legitimately deduce that his decisions were effective.

Functional Specifications for the Prototype Decision Practice System

The cognitive model described above furnished a level of detail beyond Merrill's fairly global prescriptions for USE-PRINCIPLE objectives. In essence, it provides the rationale for constructing a representation of decision problems that is "sufficient so that all of the critical aspects of the cause-and-effect relationship are represented to the student" (Merrill, 1983, p. 318). It allows the formulation of specific functional requirements that a system for decision-making practice should satisfy.

We make two important assumptions at this point. The first assumption is that a practice environment which replicates the ASOC environment with all its essential objects in a functional rather than literal sense, and which allows the student to exercise the cognitive model described above, will have a positive effect on the acquisition of the target decision-making skill and that there will be positive transfer from the skill so acquired in the practice environment to performance in the real, operational environment.

The second assumption is that practice is most effective if it is modulated in complexity and supported by instructional features like prompts (which are secondary presentation forms in Merrill's CDT) and artificial instructional feedback. Such modulations and support should occur on the basis of changes in student skill level.

Both of these assumptions can only be tested empirically. The prototype practice environment must, therefore, permit experimentation with different complexity schedules and levels of instructional support.

Together these assumptions and the cognitive model led directly to the functional specifications presented in Table 8. The functional specifications had to be tempered against the development constraints process described in the next section.

Development Constraints

The primary resource constraints were time, funding, and the hardware/software selected for implementation. The calendar time available from starting the design to evaluation was 12 months. The software to be used, Knowledge Engineering Environment (KEE), had been selected and purchased during Phase I of the project. The hardware (Table 9), two Sun 3/60 workstations, was purchased, in place, and configured to run KEE prior to starting the design.

To build a system under these constraints that meets the above goals meant that certain limitations on the capabilities of the simulation environment had to be accepted. These limitations, given in Table 10, were chosen to simplify the simulation modeling while retaining all essential FDO decision-making parameters.

Table 8. Functional Specifications for the Prototype Decision Practice Environment (SuperKEATS)

ELEMENTS	FUNCTIONAL SPECIFICATIONS	
	MACRO-LEVEL	MICRO-LEVEL
ELICITING STIMULI	PROVIDE MESSAGES THAT ELICIT CHANGES TO SITUATION ASSESSMENT AND PLANNING	PROVIDE MESSAGES THAT ELICIT DECISIONS ON AIR REQUEST DISPOSITION
DECISION PROCESS	PROVIDE ACCESS TO CURRENT OPERATING PLAN FOR REVIEW PROVIDE ACCESS TO AN INFORMATION ENVIRONMENT CONTAINING AT A MINIMUM • AIR ASSET STATUS • AIR FORCE ORDERS • ARMY CORPS GUIDANCE • INTELLIGENCE • MAP	PROVIDE ACCESS TO CURRENT OPERATING PLAN FOR REVIEW PROVIDE ACCESS TO AN INFORMATION ENVIRONMENT CONTAINING AT A MINIMUM • AIR ASSET STATUS • ARMY CORPS GUIDANCE • MAP
IMPLEMENTING RESPONSE	PROVIDE CAPABILITY TO CHANGE CURRENT OPERATIONS PLAN	PROVIDE CAPABILITY TO: • CHANGE AIR STATUS • TASK MISSIONS • REFUSE MISSIONS • REQUEST SUPPORT ASSETS
FEEDBACK (NATURAL)	PROVIDE A SIMULATION OF THE TACTICAL ENVIRONMENT WHICH SHOWS PLAUSIBLE EFFECTS OF AIR SUPPORT	PROVIDE A SIMULATION OF AIR FORCE OPERATIONAL UNITS WHICH GIVES PLAUSIBLE RESPONSES TO TASKINGS
FEEDBACK (ARTIFICIAL)	PROVIDE "BEAN COUNTING" SCORES ON OVERALL AIR SUPPORT EFFECTIVENESS PROVIDE MAP WITH AUTOMATIC DISPLAYS OF TARGETS	PROVIDE FOR TRIAL FEEDBACK PRIOR TO DECISION IMPLEMENTATION PROVIDE CRITIQUE OF TASKING WITH EXPLANATIONS AFTER DECISION IMPLEMENTATION PROVIDE CAPABILITY TO TURN THE ABOVE TYPES OF FEEDBACK ON AND OFF
PROMPTS	PROVIDE HINTS FOR PLAN CHANGES PROVIDE PLAN OPTIONS PROVIDE MEANS TO TURN THE ABOVE PROMPTS ON AND OFF *(THIS CELL NOT IMPLEMENTED!)*	PROVIDE SUGGESTIONS FOR TYPES OF AIRCRAFT TO USE, WITH RATIONALES PROVIDE SPECIFIC OPTIONS FOR TASKING PROVIDE FOR CAPABILITY TO TURN THE ABOVE PROMPTS ON AND OFF
MODIFICATION	PROVIDE AN EXERCISE BUILDING CAPACITY WHICH PERMITS GENERATION OF MORE OR LESS COMPLEX PRACTICE GAMES OR OF GAMES WHICH FOCUS ON PARTICULAR VARIABLES (AIRPLANE TYPE, BASE CLOSURE, TASKING MODE, ETC.)	
USER INTERFACE	PROVIDE CAPABILITY TO PERMIT STUDENTS TO CONFIGURE SCREEN TO THEIR NEEDS PROVIDE AN INTERFACE TO OPERATIONAL INFORMATION DISPLAYS THAT IS EASY TO LEARN AND USE, ELIMINATES TEDIOUS BOOKKEEPING TASKS AND RETAINS FUNCTIONAL RATHER THAN LITERAL FIDELITY	

Table 9. SuperKEATS Development Environment

SuperKEATS	• KNOWLEDGE BASES • LISP FUNCTION FILES • APPLICATION IMAGE (25 MB)
PROGRAMMING ENVIRONMENT	• KEE™3.1.104 • SUN COMMON LISP 2.1.3 • UNIX 3.5 (100 MB SWAP)
HARDWARE PLATFORM	• SUN 3/60 WORKSTATION • 24 MB RAM • 327 MB HARD DRIVE • 1152 x 900 COLOR MONITOR • HP LASER JET PLUS PRINTER

Table 10. Simulation Limitations

- Terrain is represented by a featureless plane. Spatial considerations include position relative to and distance from the Forward Line of Own Troops (FLOT).
- There is no weather. (Base attacks may, however, be used to close bases.)
- There are no complex strategies for Army units (i.e., no autonomous Army units).
- There are no complex Army unit movements (units move in tracks).
- There are no front-line Army unit interactions between units in the same Corps (tracks do not intersect).
- The front-line units are always in "contact" with their opposing units.
- The only types of Army units are tank units.
- The only air request targets are tanks.
- The ASOC has direct tasking authority with no interference or help from higher levels in the Air Force command and control hierarchy.
- The WOC does not optimize the allocation of aircraft to missions.

Development of the SuperKEATS Practice Environment

Development of SuperKEATS initially concentrated on the simulation portion of the system. After a rudimentary simulation was implemented so that a user could "play the ASOC game," instructional features were added. The development followed a straightforward sequence of design: implement/integrate and test. These development steps are described briefly below.

The design featured an object-oriented approach much like that of the precursor KEATS system. Objects were grouped as simulation, Army, Air Force, interface, event, and other objects. Associated with each object were its attributes, input messages, a brief description of the processing of each input, and output messages. Attributes define the state of an object. In an object-oriented paradigm, all objects communicate with each other via messages. Hence, the input messages define what an object must respond to and the output messages define what the object will send or ask of other objects.

In certain cases, the processing for a particular input message was more extensive than could be briefly described in a sentence or two. These cases were called algorithms and were written up in the detail necessary to permit implementation. In addition, because SuperKEATS would be developed using the KEE software shell, technical notes were written about how required interface features could be most efficiently implemented in this software.

On completion of the design phase, implementation started with a brief period of prototyping. During this phase, the simulation objects were entered into the KEE shell and typical displays with very limited functionality were produced for review. Prototyping provided the framework within which later development would occur, and allowed a feasibility and usability check of interface concepts.

Prototyping was followed by detailed implementation. First, the models for the Army and Air Force were developed in parallel with the interface for running a game. Then functionality was added to allow building a game and running a game in real-time. Also at this point, the instructional features for suggestions and feedback were introduced based on the rules obtained through knowledge acquisition from ASOC experts using KEATS. Added late in the development was the display sequence for game selection, assessment, planning, and setup before starting the game.

Testing occurred at intervals during the later stages of development by having a person who was not implementing SuperKEATS but was familiar with ASOC actions run the latest version, list bugs that were found, and recommend improvements. Another test was conducted by having a user-interface expert try the system and recommend improvements. SuperKEATS was first shown to members of the contracting

agency (AFHRL/LRG) for their review. Improvements, upgrades, and additions were then made before the ASOC members used the system and made their evaluations.

Two Training Systems: KEATS and SuperKEATS

Both systems support training for decision-making skills. The following description is a side-by-side comparison in which the primary focus is on how close either system comes to satisfying the instructional strategy specifications for a decision practice environment as presented in Table 1. First, the common characteristics of both systems are highlighted.

Common Characteristics. Both KEATS and SuperKEATS are prototype training systems which provide students with a practice environment for decision-making. Both present realistic, and in all essential aspects complete, microworlds to the student. They are both object-oriented; i.e., they contain objects that are replicas of real objects in the operational ASOC world, and these objects have been implemented by means of object-oriented programming. The microworlds are to a high degree isomorphic to the real world they represent.

Both systems are built around the concept of an "exercise," which is familiar to the potential users. In both cases, an exercise consists of an initial *orientation phase* and a subsequent *operations phase*. During the orientation phase, the student formulates an overall plan and puts the available resources into a corresponding state of readiness. During the operations phase, the student reacts to events as they occur by making micro decisions to deal with each specific event and by making macro decisions to adjust his overall plan to the overall course of the battle as it develops. The exercise flow in KEATS was shown in Figure 4. The exercise flow in SuperKEATS is shown in Figure 8 for comparison. The flows are very similar with the exception of some instructional elements that have been inserted into SuperKEATS. These elements are explained below.

An exercise can be of arbitrary length in either system and can therefore address short-term (hours) as well as longer-term (1 day to several days) considerations in ASOC decision-making. KEATS is more suitable for short-term exercises in that it does not depict air support effects on the ground battle. SuperKEATS was explicitly designed to overcome this shortcoming and can therefore address both short- and long-term decision problems. In SuperKEATS, exercises are called "games" to indicate to the student that the microworld he enters when he uses the system is only "real" in principle (functional fidelity) rather than in a literal sense (literal fidelity).

Finally, both systems are in essence "shells." Both contain exercise (or game) building facilities which allow the construction of arbitrarily lengthy and complex practice scenarios. The exercise building

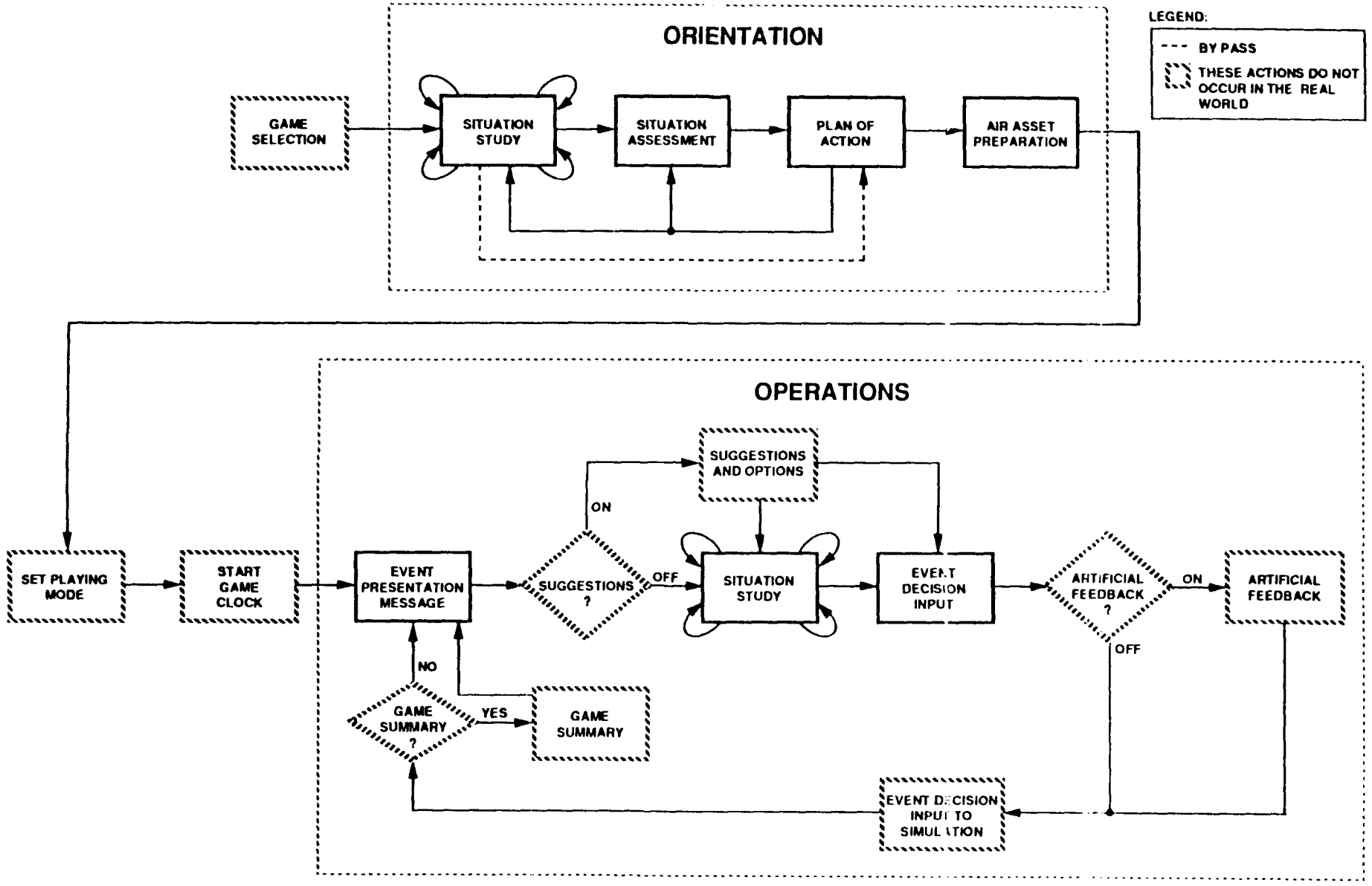


Figure 8. Exercise Flow in SuperKEATS.

facilities permit researchers and knowledge engineers[4] to modify and control the problems presented in the decision environment to test particular hypotheses or to elicit knowledge on particular topics.

Differences between KEATS and SuperKEATS as Training Systems. The fundamental difference between the two systems is that SuperKEATS allows free play whereas KEATS does not. This difference is due to the fact that SuperKEATS is based on a dynamic simulation of the ground battle where the battle is dependent on relative strengths and objectives of the ground units and on the effectiveness of air support provided by the ASOC. KEATS, on the other hand, is based on a fixed script that is almost totally unaffected by the actions of the student or player. The simulation in SuperKEATS is highly simplified but apparently quite effective.

A second, but less significant difference is that SuperKEATS has strictly functional fidelity, whereas KEATS has more literal fidelity. The overall character of SuperKEATS is therefore somewhat more abstract and more game-like than KEATS, which, at least superficially, has a more exercise-like character. This difference is underscored by the fact that KEATS is supported by color and by real maps, whereas SuperKEATS has monochromatic displays and a stylized, simplified map.

The following discussion compares the two systems in greater detail by examining how each of them satisfies the instructional strategy specifications that were developed in the preceding section on the basis of the cognitive model of the target task.

Eliciting Stimuli. Both systems present eliciting stimuli in the form of incoming messages to the ASOC on a special display that is called a "Message Desk" (KEATS) or a "Message Log" (SuperKEATS). These messages inform the student of events that have occurred in the tactical environment, of changes in fire priorities and plans made by the supported Army Corps, and of requests for air support from the various divisions.

KEATS has standard message templates for 15 types of events and in addition can display an unlimited variety of scripted free-form messages. SuperKEATS has standard message templates for 4 types of events and cannot display free-form messages. Most templated messages in either system (and the free-form messages in KEATS) elicit macro and/or micro decisions depending on the overall situation and the perception of the student.

The basic difference between the two systems is that the messages in KEATS are scripted during exercise building, whereas in SuperKEATS the messages are generated by the simulation while a game is in progress.

[4]In KEATS only.

Decision Process. Both systems support the decision process with an information environment that contains all essential information sources found in the actual operational environment, including orders (Daily Operations Order, Air Tasking Order), briefings given by the members of the ASOC team during the shift change in the morning, status boards and maps.

In KEATS, the student can engage in some limited dialogue with ASOC team members and with agencies outside the ASOC by accessing a display which shows the inside of the ASOC van and a list of the outside agencies outside of the van (literal fidelity). Such dialogue is not possible in SuperKEATS, although the student can access information on bases, divisions and corps (friendly and opposing).

As mentioned before, maps are available to the student during play with either system. In KEATS, the maps are real and are not on the system itself. Changes in the ground positions are shown by changing acetate overlays. Targets must be plotted by hand. In SuperKEATS, the map is on the system itself in a highly stylized form (a featureless grid which indicates only relative positions). Changes in the ground situation and target positions are automatically plotted by the system.

The student develops a current operations plan on the basis of a situation assessment in either system. He can access and change the plan anytime during the exercise or game. However, plan changes have no effect in KEATS, whereas they do influence system feedback to the student in SuperKEATS.

In KEATS, the display hardware limits to one display the amount of information that can be on-screen at any one time. SuperKEATS uses a large, high-resolution screen which enables simultaneous display of multiple windows. The SuperKEATS display is therefore more like the actual van environment where multiple status boards and maps are always on display.

Implementing Response. In either system, the student has a complete array of basic operational responses available to implement his decisions on either the macro or the micro level. He can task missions, refuse requests, establish smoothflow operations, divert missions, cancel missions, put aircraft on various stages of alert, change the current operations plan, and ask the Army to coordinate their actions with his. In KEATS, a large number of the micro responses are accomplished with a literally replicated tasking form that is used during actual ASOC operations. In SuperKEATS, the same actions are accomplished with more convenient forms that are different from the real format but functionally equivalent.

Feedback (Natural). *The most significant difference between the two systems is in how they react to* student decisions. KEATS can provide natural feedback only in terms of reactions from tasked wings and squadrons. Wings or squadrons will accept a tasking if they have the required airplanes, if the airplanes are at the necessary state of alert (or readiness), and if time and distance allow the airplanes to be over

target within the specified time frame. If these conditions are not satisfied, the wings will -- as in real operations -- refuse the tasking.

SuperKEATS can do all this too but, in addition, SuperKEATS displays the effect of air support on the ground battle. The effect depends on the number of aircraft that are tasked, on the relative effectiveness of the type of aircraft and its ordnance, on the relative strengths and objectives of the opposing ground units and on air support coordination with the supported units. The effect is made visible on the map by the placement of targets. When targets begin to migrate past the Forward Line of Own Troops (FLOT) into "Blue" territory, air support is not as effective as desired; when the targets migrate in the opposite direction, air support is effective and may even be too effective if the objective is to merely "hold the line."

In addition to feedback on the waxing and waning of the ground battle, SuperKEATS also provides feedback on each individual mission by displaying messages that inform the player that a mission has taken off, is over target, or is returning to base (RTB). RTB messages consist of pilot reports on mission effectiveness which indicate how many tanks were destroyed and how many aircraft were lost during the mission. No such feedback is available in KEATS.

Feedback (Artificial). KEATS does not include any artificial feedback. SuperKEATS provides artificial feedback on both the macro and micro levels.

On the micro level, SuperKEATS compares student-generated tasking responses with system-generated tasking responses. It then indicates to the student whether or not the student's response corresponds with one of the system-generated responses. SuperKEATS does not rank the quality of its own responses and is not capable of assessing whether a student response is better or worse than one of its own responses. If the student response does not correspond to a system option, SuperKEATS will supply a "critique" which indicates what option the system selected and the rationale for the selection. The student can try out a number of different responses before he decides to actually go ahead with one of them. Once he actually "sends" a response, it is inserted into the simulation and the effects are computed.

On the macro level, SuperKEATS supplies a number of overall "Game Effectiveness Measures" or GEMS. GEMS are non-judgmental, quantitative scores which reflect how effectively the available air assets were used to support the ground battle. Figure 9 shows a typical game summary display in SuperKEATS. The various effectiveness measures are explained in Table 11.

Both types of feedback (macro and micro) can be suppressed.

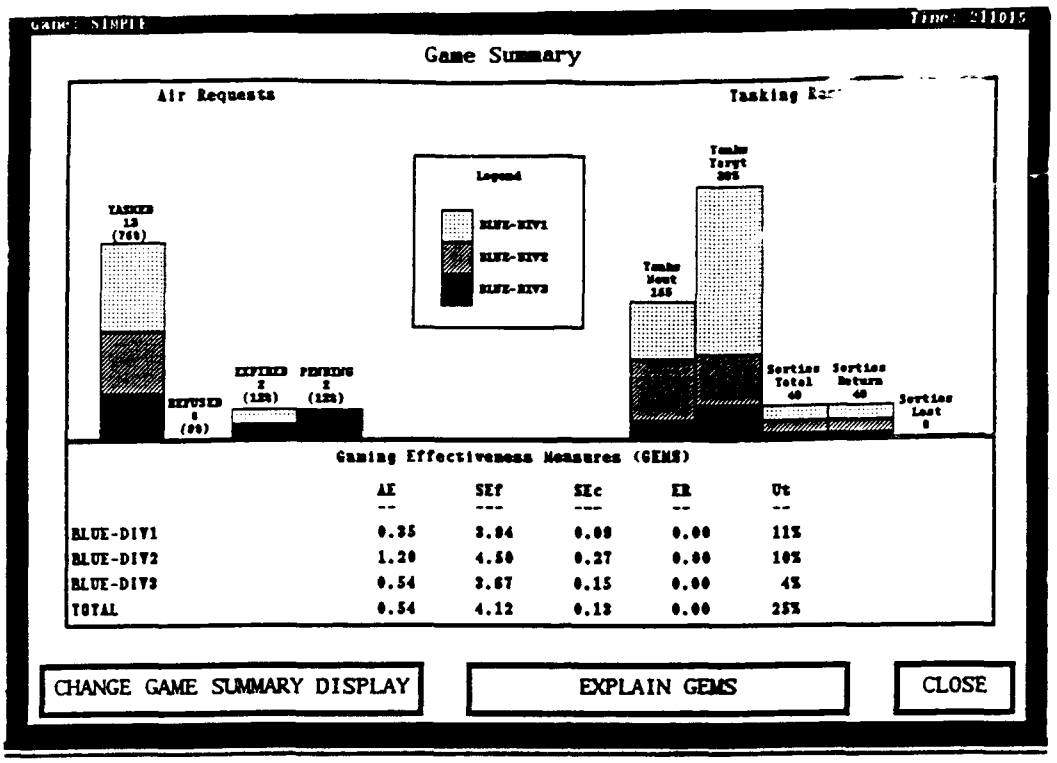


Figure 9. Sample Game Summary Display.

Table 11. Explanations of Game Effectiveness Measures (GEMS)

1. AE = Air Effectiveness = Tanks Neutralized/Tanks Targeted. The greater the value of AE, the better. AE sometimes exceeds 1 when reserves are neutralized in addition to the tanks originally targeted on the air request.

2. SEf = Sorties Effectiveness = Tanks Neutralized/Sorties Total. The greater the value of SEf, the better. This value reflects the aircraft type mix, the presence of red air defenses, and air/ground coordination.

3. SEc = Sorties Economy = Sorties Total/Tanks Targeted. A low SEc shows economical use of sorties.

4. ER = Exchange Ratio = Sorties Lost/Tanks Neutralized. A low ER is desirable. Higher ERs occur with poor mission configuration, good red air defenses, and lack of air/ground coordination.

5. Ut = Utilization = Sorties Total/Sorties Projected. Ut should ideally approach 100% at the end of the game to show full utilization of all available resources.

Prompts. SuperKEATS can provide prompts; KEATS cannot. Prompts in SuperKEATS are provided in two types, either of which can be suppressed. One type of prompt provides general guidance "suggestions" as to what type of aircraft should be used and which of several requests should receive priority. The second type of prompt provides specific tasking "options" with the number and type of airplanes available. These options are always based on the currently active operations plan.

Modification. Both systems have an exercise building capability which permits generation of more or less complex practice exercises or games, and/or games that focus on particular variables that must be taken into account during decision-making.

The KEATS exercise building mode is more convenient and less limited than that of SuperKEATS. In KEATS, an exercise author can create any number of tactical objects from the existing classes of objects and specify their editable characteristics. Events can be created either from editable event object classes or as free-form messages. There is essentially no limit to the complexity and "trickiness" of the exercises that can be generated.

In SuperKEATS, the exercise author can only change the initial conditions by changing a limited set of parameters related to ground units, airplanes, bases and simulation parameters. Currently, SuperKEATS contains three games at three different complexity levels. Table 12 lists the differences in complexity parameters among these games.

In addition to the parameters listed, the SuperKEATS exercise builder can regulate the length of a game, its start time and the frequency of simulation updates. Games can be mini-games covering a single time block or a full game covering an entire day. The exercise builder can also change fire priorities, ground unit objectives, number of tanks in front-line units and reserve units and the number of wings (up to 10) and squadrons (up to 3 per wing).

Finally, in SuperKEATS game speed can be regulated. Games can be played either in a pushbutton mode where the player determines when the next event is called, or in an automatic mode where events occur either in real-time, twice as fast as real-time or 10 times as fast, depending on what the player has selected. KEATS has only one speed.

User Interface. Considerable effort has been expended to make the user interface in either system as convenient and easy to learn as possible. Both interfaces use mouse as well as keyboard inputs. The major difference between the two systems is that information access in KEATS is sequential with one display at a time, whereas SuperKEATS allows the user to configure the screen to his personal needs with individually resizable and movable windows.

Table 12. Game Complexity Parameters

Parameter	Effect	GAMING LEVELS		
		Simple	Intermediate	Complex
Blue Forces Air Status	Base air attacks	No base air attacks	Occasional base air attacks	More frequent base air attacks
Small Unit Attack Capability	Base ground attacks	No base ground attacks	No base ground attacks	Base ground attacks
JAAT Capability	Type of requests Air support effectiveness	No JAAT	No JAAT	JAAT supported
Red Air Defense	Aircraft losses	No aircraft losses	Aircraft losses simulated	Aircraft losses simulated
Number of WOCs	Aircraft types	1	3	6
Base Defense Forces	Base attack effectiveness	Good base defense simulated	Good base defense simulated	Effectiveness of base defense varies
Number of Turns	Asset use/ availability	All assets equally available	Asset availability varies	Asset availability varies

JAAT = Joint Air Attack Team

Status boards are updated automatically in both systems, but only SuperKEATS provides automatic map updates. The objects that are displayed on the map in SuperKEATS can be "moused," which results in the display of a window with textual information.

Summary. The two systems address the same cognitive task but with vastly different capabilities. KEATS is more flexible and has a more "operational" *appearance*, but it has the fundamental shortcoming of very limited feedback. SuperKEATS, on the other hand, is much more "operational" in function because it provides a fully reactive microworld which includes credible natural feedback and controllable artificial feedback. SuperKEATS is in essence a full realization of the functional specifications for the practice module of a full-scale DTS, whereas KEATS is not. KEATS can probably be effective if it is used with a human instructor who can provide the necessary feedback and if the exercises are very short in

duration (i.e., "snapshots"). SuperKEATS can be effective without a human instructor. Also, the longer a game runs in SuperKEATS, the more apparent will be the effects of earlier decisions, especially decisions made on the macro level.

One very interesting feature of SuperKEATS is the fact that it can be played purely on the macro level. The micro decision level can be completely eliminated by establishing a mode of operation called "smoothflow" for all squadrons. In smoothflow, squadrons simply send out missions at regular intervals to some designated division. Once smoothflow is established, the ASOC is essentially out of the game; i.e., air support is "on automatic." The player can then select a game speed of 10 times real-time and observe the effects of his macro allocation of airplane assets to divisions. He can also change the macro allocation in midstream if the battle is not going to his satisfaction.

Evaluation

SuperKEATS was evaluated on only one occasion. One ASOC expert and one novice spent approximately 7 hours each with both simple and complex games. Detailed protocols were collected automatically. At the end of the day, both subjects filled out questionnaires.

The evaluation produced highly different game results for the two different experience levels (novice and expert). These results are shown in Table 13.

The questionnaire responses showed that SuperKEATS was judged to be useful to the ASOC, with an overall rating of 4 on a 5-point scale. The expert particularly liked the ACTION - REACTION - RESULT chain introduced by SuperKEATS, underscoring the achievement of the major SuperKEATS objective to furnish a reactive environment which was lacking in KEATS. Both expert and novice disliked display functionality, recommending more compact and complete information displays. Both subjects encouraged further development and suggested use of SuperKEATS in operational exercises.

III. DISCUSSION

The overall goal of the present effort was to develop prototype training packages for tactical decision-making tasks that are more affordable, more accessible, and more effective than the training currently provided by large-scale exercises. The intent was to contribute to, and hopefully advance, the state of the art in an emerging technology of desktop-based training environments for decision-making which began to emerge during the early 1980s in response to a widely recognized need for more accessible and economical means to train tactical decision-makers in all military services. As it stands today, such a technol-

Table 13. Gaming Measures and Division Movement Comparisons

	Novice After 119 Minutes	Expert After 141 Minutes	Expert After 249 Minutes
Number of Requests	10	14	25
Tanks Neutralized	44	160	279
Tanks Targeted	200	299	634
Sorties Over Target	44	88	144
Sorties Lost	2	5	11
Air Effectiveness (AE)	0.22	0.54	0.44
Sorties Effectiveness (SEf)	1.00	1.82	1.94
Sorties Economy (SEc)	0.22	0.29	0.23
Exchange Ratio (ER)	0.05	0.03	0.04
Utilization (Ut)	10%	19%	31%
Enemy Advances in Kilometers			
Division 1	10	1	1
Division 2	0	0	0
Division 3	0	0	0

ogy is no longer as hampered by either hardware or software constraints as it was at its beginning and as it still was at the start of this project. The primary obstacle today is the ability of instructional designers to design effective and efficient training treatments and to develop a given training design into a fieldable training package at an affordable price.

The project reported here addressed training design and development issues directly by formulating the problem of training decision-making skills as an instructional design problem in which the solution for the two variables content and strategy had to be determined. Accordingly, the specific objectives for the project were to find means to externalize the task knowledge deployed by experts during decision-making and to use this task knowledge in the development of training methodologies that would reliably lead to the acquisition of decision-making skills. To the extent possible, the methods to be developed were to be cost effective.

The first prototype system developed under this project, KEATS, incorporates in one system a methodology to acquire expert decision task knowledge as well as a methodology to train decision-making tasks. The second prototype system, SuperKEATS, represents strictly a methodology to train decision-

making tasks. These two systems are the products of a development methodology which was intentionally evolutionary and economical, but which was also, to some extent, dictated by concerns other than cost effectiveness.

The questions that arise at this point, the end of the project, are the following:

1. What are the technical strengths and weaknesses of the methodologies and prototype systems that were developed?

2. What can be said about the cost effectiveness of the methodologies that were developed and the development method itself?

3. Should this work be continued, and if so, what is a reasonable program that would lead to a point where significant cost savings could be achieved by fielding systems on a broad scale?

The discussion below deals with each of these questions in turn but it must necessarily be prefaced with a caveat: Since only very limited evaluations have been performed up to this point, there are no reliable data that would permit definitive assessments of the effectiveness of either the knowledge acquisition methodology or the training methodologies represented by the two prototype systems.

Technical Discussion

Computer-Aided Knowledge Acquisition Methodology

The knowledge acquisition methodology which evolved in the course of this project consists of the following steps:

1. Initial surface knowledge acquisition through study of available documentation, followed by focused and structured case-based interviews using paper-based hypothetical scenarios.

2. Development of a computer-based scenario generator (KEATS) on a microcomputer using object-oriented software (Smalltalk) and rapid prototyping techniques.

3. In-depth knowledge acquisition using computer-presented decision scenarios generated by experts, automatic knowledge elicitation queries, on-line entry of query responses by experts, and automatic generation of detailed decision-making protocols.

4. Protocol analysis by knowledge engineers leading to the formulation of production rules which are edited into final form by the same experts who produced the protocol.

Limited applications of this methodology have shown that Step 3 leads to a very rich protocol, and that Step 4 is a fairly straightforward exercise which requires a minimum of inference or interpretation on the part of the analyst/knowledge engineer. The richness of the protocol appears to be a function of the degree of realism inherent in the decision scenarios that are presented to the expert and the degree of realism built into the user interface: The expert must be able to behave and respond in a manner that very closely approximates actual operational behavior and responses. The overall impression gained thus far is that this methodology is very productive; i.e., that it yields a high return in terms of well-formulated knowledge for the expert and analyst manhours spent in Steps 3 and 4.

This high return is, of course, bought with a relatively high investment of manpower in Step 2. Whether that investment is justified depends not only on how productive the method is, but also on whether the method overcomes the problems that gave rise to its development and on how well the method generalizes to other knowledge engineering requirements.

Given the limited empirical data regarding the efficacy of KEATS as a knowledge engineering support tool, it is not possible to state unequivocally whether the problems that gave rise to its development have indeed been solved. KEATS does present robust cases for knowledge engineering. The environment it presents to experts is convincing enough to fully engage their operational decision-making skills. However, there are no definitive answers on the CATCH 22 problem, although there is good reason to believe that KEATS can have a facilitating effect in this regard. There are no final answers on the access and the motivation problems either. The system has been at the ASOC for more than a year, but it has been used very little during that time.

In view of the tentative and incomplete results available to date, it may be too early to deal with the issue of generalizability to other domains. However, the prototype concept was confirmed as basically sound by a knowledge product which was substantial for a first formative trial. Some speculative extrapolation to other domains appears permissible and useful at this point.

KEATS was designed for the ASOC domain, which was identified as a domain that is representative of TC^2 in the Air Force. There is no apparent reason why KEATS could not be expanded to include all the objects that the larger world of an Allied Tactical Operations Center (ATOC) includes. By the same token, there is no reason why the same technology that was used in KEATS could not be applied to a Wing Operations Center (WOC) or a Tactical Air Control Party (TACP).

The key to the issue of transferability is in the modeling requirements for decision problem presentation and for decision implementation. As long as decision input and output for a domain are in the form

of verbal messages, the KEATS technology is directly applicable. It would not be directly applicable, for example, to the decision problems in an airplane or in a nuclear power plant. In those instances, decision input (problem presentation) is in the form of complex visual images and dial indications. Decision output is in the form of direct control manipulations. The control manipulations must be followed by more or less instantaneous feedback. A full simulation of the physical system is required in these cases. Verbal decision input and output are common across the domain of Air Force TC^2; therefore, KEATS is at least transferable to other Air Force TC^2 nodes. To the extent that the input/output conditions are satisfied, it is also directly applicable to other decision environments.

Direct transfer or direct applicability is understood quite literally. The KEATS prototype consists of domain- or application-specific code and generic code. The generic code (object classes) represents about 70% of KEATS, and it is that portion that can be directly transferred to another domain. The objects of the new domain must be added and their behavior coded as methods. The latter is a nontrivial undertaking but the object-oriented fast prototyping environment offered by Smalltalk reduces labor requirements. The combination of reusable code and fast prototyping technology makes transfer to other domains an economically attractive undertaking.

Cost effectiveness of transfer may, however, be a moot point if there is no other way to acquire the knowledge. Our evidence suggests that knowledge acquisition for decision-making in a "messy" domain could not be done effectively, if at all, with interview-type techniques. KEATS provides the necessary structure and precision for effective expert-knowledge engineer communication and prevents the type of chaos that can ensue from imprecisely communicated and understood questions and answers.

More than that, a KEATS-type system grows, in the course of its application as a knowledge engineering system, into a precursor or prototype of the target training system. In this project, KEATS has provided a baseline for what the final training system should look like and do. Our general approach to this project has therefore become an evolutionary rather than a serial approach. It is interesting to note that a similar basic approach is currently being pursued by at the Army Research Institute (Stoddard et al., 1986), where a first training system prototype is being used to assist in the acquisition of knowledge for a cognitive skills tutor.

Training Methodologies

The second objective of this project was the development of technologies for training decision-making skill which were to be tangibly instantiated with training system prototypes. The two training system prototypes that were developed by this project are KEATS and SuperKEATS. To reiterate: KEATS is, on

the one hand, a system that aids in the acquisition of instructional content required for building training system prototypes; and it is also a training system prototype.

Training Methodology Instantiated in KEATS. KEATS has not been subjected to a formative evaluation as a training system. Thus, data which could provide clues as to its efficacy as a training system are not available. However, the technical features and characteristics of KEATS are known and enable speculation on the subject.

Basically, KEATS is capable of presenting practice stimuli in the form of decision problems, of enabling realistic information search activities, and of accepting decision responses. KEATS is, however, deficient in the area of feedback. Taylor (1983), in a review of literature relevant to unaided decision-making, noted very accurately that developers of decision-making training programs face two problems: "providing for consequences of decisions" and "evaluating decision-making performance." KEATS does both, but only to a limited extent.

KEATS provides for consequences of decisions by manipulating airplane availability and by adjusting the script. Airplanes which are enroute can be tasked only by diverting them. Attrition occurs as a function of threat. KEATS provides for realistic consequences with respect to the air side of the battle, but it does not provide for decision consequences in terms of the ground war. The learner's decisions will have no influence on whether a scripted advance or retreat of opposing ground units takes place or not. He will therefore never know whether his manipulations of the supply side of the equation had any effect on the demand side. The features of KEATS which provide for consequences on the air side contribute much to the knowledge engineering function by making cases more robust (i.e., impervious to deviations from the script). They are also required for training but, by themselves, they provide a skewed, one-sided picture.

KEATS also evaluates decision-making performance but, again, only to some extent. Nickerson and Feehrer (1975) were adamant in suggesting that decision-making performance should be evaluated in an a priori fashion; i.e., on the basis of how well the decision-maker used available information at the time the decision was made. They felt that an *a posteriori* evaluation (i.e., an evaluation based on the effects of the implemented decision), especially in complex domains, could not be accomplished at all in that the effects are inevitably confounded by factors not under the decision-maker's control or cognizance. KEATS does perform a partial *a priori* evaluation of a decision. It checks whether the decision would violate any physical constraints and it does provide appropriate feedback. KEATS thus examines the technical feasibility of a decision, but it has nothing to say about its tactical wisdom. If KEATS is used with

an instructor or coach, the latter can provide that missing aspect of feedback. In the instructor-assisted mode, KEATS can provide a full measure of a priori decision evaluation (if the instructor is qualified).

Thus, KEATS is not yet the training system to which this project was aiming. It does satisfy the requirement of running on low-cost hardware.[5] However, it still requires an instructor for decision-making training. If, and when, KEATS reaches full fidelity, it will be a fully functional procedures trainer in the standalone mode. Its adaptability to local needs, particular scenarios, and individual trainee requirements -- as afforded by the BUILDING mode -- makes it more flexible and economical than the paper-based System Training Exercises which are currently used by the ASOC and other TACS nodes. However, as a decision training system, KEATS is merely a beginning.

Training Methodology Instantiated in SuperKEATS. SuperKEATS was built to overcome KEATS' limitations in the area of feedback. It is discussed here from the perspective of how it relates to other technologies for training decision-making skills, such as the earlier work on small-scale, inexpensive, decision training systems referenced in Section I and to Intelligent Tutoring Systems.

In terms of scope and objectives, only SuperKEATS has had training as its main objective. It provides training on the macro level by allowing modification of exercise complexity, and on the micro level by using prompting and artificial feedback. The work of Madni et al. (1987) focused mainly on the modeling of an intelligent adversary and allowed modification of the model as a means to train various objectives. Obermayer et al. (1984) showed that modeling with an expert system could reduce the labor intensity of training, but the "Game Environment Simulator" they built was a simulation and not intended as a training system. Wilson (1982) created a wargame that might have training utility, but only as a secondary feature. Stoddard et al. (1986) had a long-term goal of producing an intelligent tutoring system for tactical decision-making but, at this stage, they report only the design for a knowledge extraction tool, which serves a purpose much like KEATS, but without explicit training features.

In terms of the domain, SuperKEATS attempted to model a more complex domain than any of the earlier systems, although it does impose limitations on weather modeling and terrain. SuperKEATS needed to model Army unit actions from the corps level down to the front-line company, Air Force units from the wing operation center to individual aircraft, and the interaction of air units attacking ground targets. Madni et al. (1987) simulated a one-on-one interaction of two surface ships. Obermayer et al. (1984) had a complex environment with ships, a helicopter, a submarine, sonobuoys, torpedos, etc., but the game was based only on anti-submarine warfare (ASW) concepts and did not attempt fidelity to a real

[5]Standard squadron-issue microcomputers need to be upgraded from 640K bytes to 4M bytes.

ASW environment. Wilson (1982) modeled only the air battle and Stoddard et al. (1986) were interested in the Army armor officer directing platoons of tanks in a land battle.

SuperKEATS, then, advances the technology base for decision-making training by tackling a complex domain intent on producing a training system with real-world operational use, where completeness and fidelity are key issues. That it has succeeded is evidenced by the positive responses of the ASOC members who have used it.

SuperKEATS is not, and was not intended to be, an intelligent tutoring system (ITS). Because SuperKEATS does provide an automated environment for practicing (and learning) decision-making skills, a comparison with an ITS puts the capabilities of SuperKEATS in perspective. In this discussion, the ITS for a gaming environment (Burton & Brown, 1982; Goldstein, 1982) will be used for comparison.

Typically, an ITS will have a model of an expert which generates responses and solutions that are compared with the student's responses. Though a standalone expert model was not built for SuperKEATS, rules of thumb and generation of tasking options are implemented that use rules obtained from ASOC experts. A key difference between the complex ASOC environment and most previous ITS environments is the existence of multiple acceptable solutions for air asset tasking. It was important for prompting that all reasonable options be presented and, for feedback, that choosing a reasonable option be accepted. Hence, an expert ASOC FDO model, as such, that always arrived at a specific tasking solution, was not necessary or useful for training in this environment.

SuperKEATS does not provide a tutor/instructor model as would be present in an ITS to direct, at the macro level, the topics of instruction, and at the micro level, to set the level and amount of prompting and feedback. SuperKEATS also does not provide a student model (as expected in an ITS) to keep track of the student's current level of expertise and topics learned. SuperKEATS is basically a practice environment which can be incorporated in a larger decision training system that also includes tutoring and student model capabilities.

A major concern with any automated system is its brittleness as the world for which it was built changes. To try to build an automated system that can cope with any changes in its area of operation would be very expensive, if even possible. If the environment is changing so rapidly that between the time the system is designed and completed the system has become obsolete or at least needs major upgrades, then building the system in the first place must be questioned. Among the major concerns with ITSs is how difficult they are to build and then how easily they fail if the environment changes. Because SuperKEATS was built as a simulation practice environment and is an alternative to a traditional ITS, its brittle-

ness is of interest. With SuperKEATS, brittleness depends on the type of change involved. Various possibilities are discussed below.

Changes to equipment such as aircraft or ordnance that affect only model parameters can be easily accommodated in SuperKEATS. The changed parameters are readily modified in the "fact" knowledge base. If equipment changes affect the suggestion/feedback rules about when to use the equipment, then changes to the rules will need to be made. These rule changes must be based on expert knowledge and will likely be somewhat involved.

Changes in the ASOC tactics that have been captured in the suggestion/feedback rules will likewise require changes in the rules. If the change is fairly basic, then the rules may need be acquired anew, possibly by using KEATS. If the change is minor, it may affect the conditions and actions of only a limited group of rules.

Changes to the functions of the command units (WOC, ATOC, Army G-3, even the ASOC) will typically have a major effect on the models of those units. Because the models are built to perform specific functions under certain conditions, changing those functions could conceivably require a redesign and redevelopment of the model. At the very least, the code controlling a particular function will have to change.

In summary, then, the more stable the environment is, the less it will cost to maintain a SuperKEATS system. Every system that works in the real world where changes occur will require maintenance and upgrades. There is no evidence that a simulation practice environment will necessarily require less maintenance than a traditional ITS.

The modeling in SuperKEATS should be easily transferable to other positions in the ASOC such as G3-Air, Intel, or Director. Because the members of the ASOC function as a team, they would all receive the same briefings as the FDO and be concerned with the same air requests. The Intel and G3-Air decisions about air requests are not concerned with tasking specific air assets but are accept/reject decisions based on request validity. There would need to be additional modeling so that spurious requests were generated occasionally.

The ATOC also makes decisions about supply and demand of aircraft; in certain theaters, the ATOC even subsumes the ASOC role of tasking air assets. Hence, the decision-making about close air support (CAS) built into SuperKEATS should be directly usable to ATOC personnel making those same decisions. If it was desired to task battlefield air interdiction (BAI) as well as CAS, then the modeling would need to be expanded.

The Army modeling in SuperKEATS is designed such that Army units request support when objectives are not being met. Transfer to other decision environments without the need for Army requests would require different models. For example, a node for air defense would require models of enemy air units launching attacks and models for results of air encounters.

In the evaluation data presented earlier, two items that particularly stand out are: (a) the large difference between expert and novice performance; and (b) the feeling expressed by both expert and novice that suggestions and feedback are very useful for training.

That SuperKEATS is clearly responsive to differences in expertise was shown by comparing the results of the ASOC expert and novice on the complex game. This is illustrated by Figure 10, where the enemy has penetrated 10 kilometers into Blue territory in 2 hours with novice support, but in 4 hours with expert support, the enemy has been held to within 1 kilometer of Blue territory.

Although the effectiveness of the suggestions and feedback have not been tested, the enthusiasm of the ASOC personnel for this aid to training is noted. The novice thought that the feedback was the best feature of SuperKEATS and suggested even more extensive suggestion and feedback information. The expert thought that feedback was important for the training of novices and for the practice of experts. He also thought suggestions would be useful for real-world (i.e., combat) tasking.

Development Methodology. Although the design and development of SuperKEATS occurred over a relatively short period of time (1 year), the complete development effort included a number of prerequisites that had to be accomplished first. The first and most obvious prerequisite was to acquire knowledge of the ASOC domain. Designing SuperKEATS required knowing what people, organizations, and things affect ASOC decision-making during a battle and how these objects interact with the ASOC and among themselves. This knowledge acquisition work on the ASOC domain was the focus of the second phase of this project (Brecke et al., 1989) and occurred over a period of about 1 year. The result of this phase was the knowledge acquisition methodology discussed above, which included KEATS. The general model of ASOC decision-making represented by KEATS and the more specific models of entities in the ASOC world that were incorporated in KEATS became the starting point for the models in SuperKEATS.

As the design of SuperKEATS proceeded, it became clear that, for realistic modeling, knowledge of the operations of other decision-making organizations in the immediate ASOC environment was important. For example, the model of a WOC had to respond to ASOC tasking requests. To know how a WOC would respond requires a very detailed understanding of how a WOC operates. This level of detail would also be needed for the Air Liaison Officer (ALO) submitting air requests to the ASOC and assigning

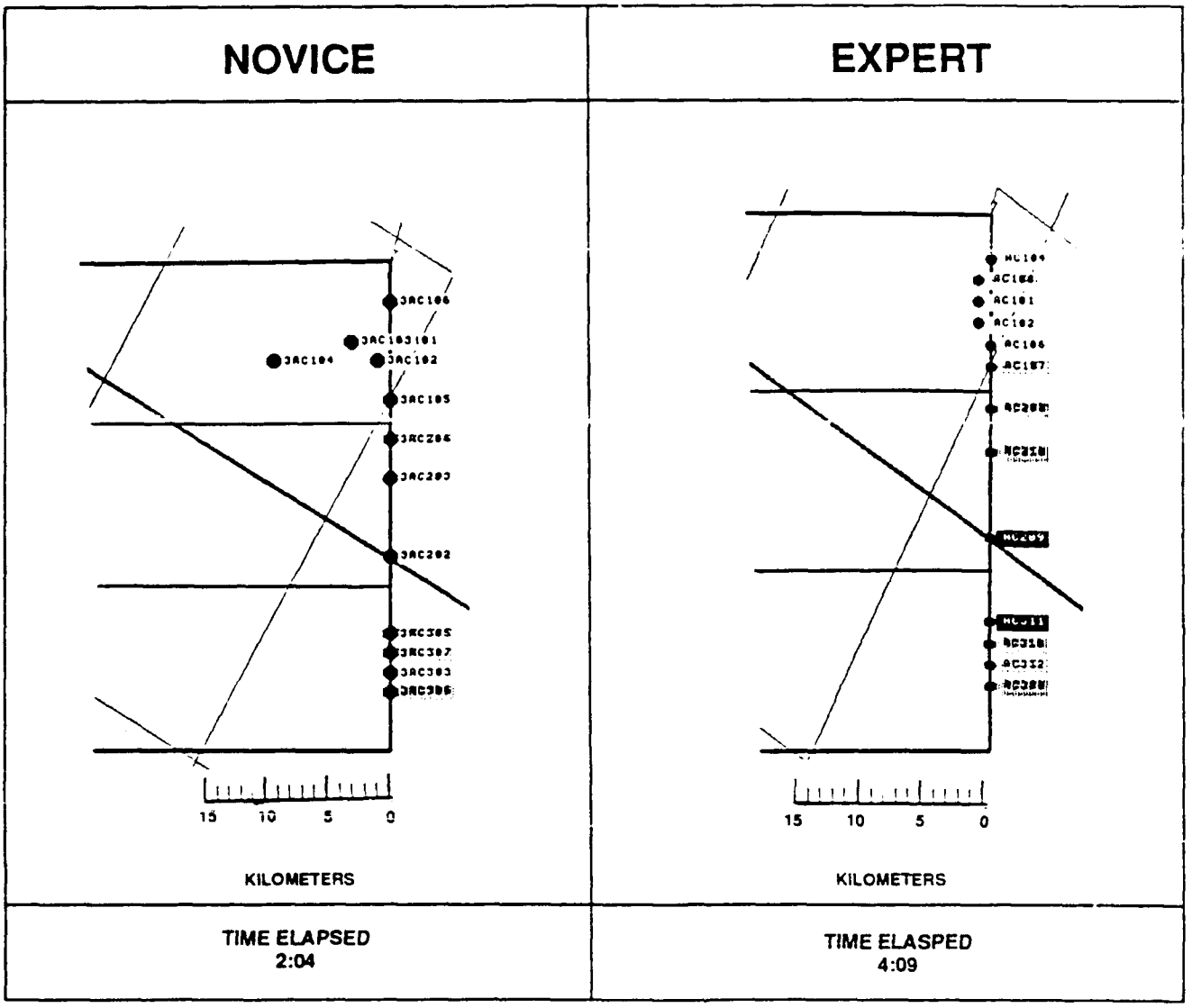


Figure 10. Results of Expert Versus Novice Tactical Decision-Making.

smoothflow missions to requests, and for corps commanders changing air support fire priorities. Because a WOC expert was unobtainable, the knowledge of how these decision-making nodes operated came either from what the ASOC members knew and expected of a WOC or from making best guesses.

Assumptions had to be made as to how much fidelity was necessary for modeling all other objects in the environment. The principle followed here was to begin with the simplest possible object modeling and to increase complexity only in response to expert feedback. This resulted in a number of simulation limitations (shown in Table 10) which were chosen to simplify the modeling of Army units such that only actions important to the ASOC would result.

As a final note on prerequisites, it is suggested that having an in-house expert (e.g., a retired ASOC member) available during the building of SuperKEATS would have resolved many "developer best guess" problems. That SuperKEATS was built without this help, and was readily accepted by ASOC personnel, shows that it can be done successfully with very limited access to the experts. However, the product can be no better than the knowledge it contains. It will probably require one or more iterations of expert-use/ model-revision to correct developer misconceptions. Having an expert always available would surely have speeded or eliminated the iterative process.

During a multi-year project, there are likely to be changes in both the initial concepts (although not the goals) and the available tools. The initial concept for this project was based on the assumption that training of decision-making in a tactical environment requires some kind of an intelligent tutoring system (ITS). As instructional design theories were studied and as results from existing ITSs became available, this concept changed to one of developing, as the product of this effort, a simulation practice environment.

During the early stages of this project, the need for a fast prototyping environment was recognized but it was also considered that a rule-based expert system would be required in an ITS-type training system. Based on these considerations and on the need to gain familiarity with the software, the Knowledge Engineering Environment (KEE) was purchased during the first phase of the project.

During the second phase of the project, when knowledge acquisition was the area of activity, the KEATS system was built. Because KEATS had no requirement for an expert system but did have the goal to place the system physically into the ASOC for their use, it was decided to use the object-oriented Smalltalk V, which had just become available and ran on PC hardware. KEATS was developed successfully in Smalltalk V, which proved to be an excellent environment for rapid prototyping. The only drawbacks were that response time was somewhat slow on AT (80286-based) machines (but 80386 machines

were significantly better) and the medium-resolution graphics limited the amount of information that could be on the screen at one time.

When the third phase started, it was recognized that using software already developed in KEATS would represent cost savings for SuperKEATS. However, the necessity to write rules, and the investment in KEE and the hardware to run it, led to developing SuperKEATS in KEE. Though KEE did have an object-oriented environment, a nice graphical interface, and an easy-to-use knowledge base editor with capabilities for multiple inheritance, it was not well suited for a simulation environment for these reasons:

1. The simulation was procedural and algorithmic while KEE is based in Lisp, which is a symbolic language. Hence, writing the algorithms was clumsy.

2. The programming support tools (such as the editor and debugger) for the Lisp code were poor compared to Smalltalk V.

3. Garbage collection was to virtual memory on disk and hence very slow -- typically on the order of minutes. This is not acceptable in a real-time interactive environment.

4. Memory fragmentation occurred after running an exercise for several hours and this slowed down system response considerably. Each workstation was configured with 16 MBytes of main memory, but for the final evaluation, 8 MBytes of storage moved from one system to the "demo" system, giving it 24 MBytes (the workstation limit). Even then, later in the exercise the expert found system response to be slower than desired.

5. The rule system proved to be awkward for the kind of rules needed for suggestions and feedback. Many rules involved comparisons of values (number of tanks, number of aircraft, etc.) and this required going into Lisp code. The syntax of the rules probably could not be read and understood by an expert.

The objected-oriented approach to building a decision-making environment proved to be very effective in SuperKEATS, as it was in KEATS Despite the drawbacks to rule implementation in KEE mentioned above, rules were a simple and efficient way to represent ASOC tasking knowledge within SuperKEATS. Rules typically reason about the current state of specific objects and were easily added to the object-oriented simulation environment.

Given the development lessons learned during this R&D project, it is recommended that similar future efforts be executed in a single software/hardware environment such as, for example, Smalltalk 80 running on PC or AT hardware. The entire development process would thus include the four-step knowl-

edge acquisition method discussed above, which would be followed by a phase where the knowledge acquired via the scenario generator is successively "plowed back" into the scenario generator and into a simulation model that is gradually added. The training environment would thus gradually evolve from the system's use as a knowledge acquisition tool.

This then is the basic methodology for developing training for decision-making tasks in tactical domains that has emerged from this project. The major features of this methodology are summarized as follows:

1. Conventional task analysis methods enable instructional designers to identify a decision task, but they do not enable a detailed analysis including a breakdown into smaller task elements.

2. Knowledge engineering methods, when supported by a computer-based scenario generator, scenario presentation and query mechanism (such as KEATS), enable the instructional designer to construct a valid cognitive task model and to acquire the task knowledge employed by expert decision-makers. This task knowledge is required for the development of the presentation elements of instruction and for the generation of practice feedback.

3. The computer-based scenario generator, initially used for knowledge acquisition, serves as a temporary training system. A copy of it is gradually transformed into the final training system by adding a simulation of the tactical world with which the decision-maker interacts and by adding knowledge-based facilities which are capable of supplying prompts and (artificial) feedback to the student.

The methodology addresses instructional strategy issues only to the extent that it specifies that the practice and test elements should conform to the cognitive task model. It leaves unanswered what a complete instructional strategy for decision-making objectives should look like; i.e., it does not address how instructional content is to be presented, when and in what form artificial prompts and feedback should be administered, and how practice should be modulated in complexity and difficulty in step with student performance.

Any number of reasonable and intuitively persuasive instructional strategies can be constructed a priori on the basis of existing instructional design theories. Training systems built through application of the methodologies described in this report and summarized above can be used as testbeds for such strategies. It would appear, though, that any candidate instructional strategy to be tested should have a high degree of *a priori* face validity, preferably higher than that achievable by merely applying an existing instructional design theory (for example, a combination of Reigeluth's Elaboration Theory and Merrill's

Component Display Theory) to the design problem. Ultimately, higher face validity can come only from a better understanding of the skill to be taught, the process of skill acquisition, and the means of control over the skill acquisition process that are available. The current literature on novice-expert differences certainly provides well-validated insights on the nature of performance differences at the beginning and end points of a skill acquisition process. What is much less clear even today is how the change from novice to expert performance (i.e., learning) comes about and how that process can be optimized by instructional manipulations and interventions. The ultimate solution to the problem of designing comprehensive instructional treatments for decision-making skills is therefore still awaiting the results of basic empirical research yet to be done. The point of view advocated here is that the work performed in this project can provide a starting point for basic research, with a very clear focus on tactical decision-making skills. SuperKEATS provides decision task practice that is designed to model the cognitive task. This practice environment can be manipulated in a systematic fashion and it can be embedded in various types of comprehensive instructional treatments.

Cost Effectiveness Considerations

Though several references to cost effectiveness have already been made in the preceding technical critique, the subject has not been discussed from a more global viewpoint. It is probably quite clear from what has been said so far that the development of a SuperKEATS-like training environment involves nontrivial expenditures, even if a streamlined development process (such as the one sketched out in the paragraph above) is used. The question is, therefore, whether a SuperKEATS-like training environment affords a better cost/benefit ratio than other currently used or potentially usable forms of training.

Because neither actual cost nor instructional effectiveness data were available, a simple parametric analysis was performed. Fourteen currently used and potentially usable training methods were rated on a scale from 1 to 10 on eight Cost (C) parameters and eight parameters that were assumed to contribute to Instructional Effectiveness (IE). Results are shown in Table 14.

The training methods were grouped into Unguided and Guided Methods. This distinction highlights the fact that current Field Training Exercises (FTXs) and Command Post Exercises (CPXs) lack the systematic guidance afforded by some plan of instruction that is designed to lead a learner from some expected initial state at the beginning of an exercise to some desired terminal state at the end of the exercise. If either KEATS or SuperKEATS were used without such guidance (for example, if either would have only one exercise or game to play), it would fall into the unguided category. That is why these forms of KEATS and/or SuperKEATS were included under Unguided Methods.

Table 14. Ratings of 14 Methods to Train Decision-Making Skills on Eight Cost and Eight Instructional Effectiveness Parameters

TRAINING METHODS	COST PARAMETERS								INSTRUCTIONAL EFFECTIVENESS PARAMETERS										
	HARDWARE ACQUISITION	HARDWARE MAINTENANCE	COURSEWARE / EXERCISE DEVELOPMENT	COURSEWARE / EXERCISE MAINTENANCE	INSTRUCTOR COSTS	CONTROLLER & OVERHEAD PERSONNEL COSTS	TDY COSTS	RESOURCE CONSUMPTION	REALISM OF INTERACTION	FREE PLAY INTERACTION	NATURAL FEEDBACK	ARTIFICIAL FEEDBACK	INDIVIDUALIZED TRAINING	EXPLICITNESS OF CONTENT	EASE OF ACCESS	COVERAGE OF TRAINING	COST SCORES (-)	INSTRUCTIONAL EFFECTIVENESS SCORES (+)	TOTAL SCORES
UNGUIDED METHODS																			
1. FTXs	N/A	N/A	10	N/A	N/A	10	10	10	10	2	7	1	1	1	1	6	40	29	-11
2. CPXs	10	10	9	10	N/A	9	10	5	9	3	8	1	1	1	3	5	63	31	-32
3. WARGAMES	9	9	6	9	N/A	7	10	2	6	10	10	1	1	1	3	6	52	38	-14
4. KEATS	2	1	3	4	N/A	N/A	N/A	N/A	4	2	1	2	4	1	10	3	10	27	+17
5. SuperKEATS	3	2	4	5	N/A	N/A	N/A	N/A	3	10	7	8	7	3	10	4	14	52	+38
GUIDED METHODS																			
6. LECTURE	N/A	N/A	1	1	10	N/A	2	N/A	N/A	N/A	N/A	8	1	10	3	4	14	26	+12
7. LECTURE + EXERCISE	1	1	2	3	10	N/A	2	1	1	1	1	10	2	10	3	5	20	33	+13
8. STE	N/A	N/A	3	2	1	5	N/A	1	2	1	1	2	2	5	10	4	12	27	+15
9. CONVENTIONAL CAI	1	1	3	3	N/A	N/A	N/A	N/A	N/A	N/A	N/A	2	4	10	10	4	8	30	+22
10. KEATS + GUIDE	2	1	3	4	1	1	N/A	1	4	2	1	2	4	1	10	3	13	27	+14
11. SuperKEATS + GUIDE	3	2	4	5	1	1	N/A	N/A	3	10	7	8	7	3	10	5	16	53	+37
12. KEATS + CAI	2	1	5	4	N/A	N/A	N/A	1	4	2	1	2	4	10	10	6	13	39	+26
13. SuperKEATS + CAI	3	2	6	6	N/A	N/A	N/A	N/A	3	10	7	8	7	10	10	10	17	65	+48
14. FULL DTS	3	2	7	7	N/A	N/A	N/A	N/A	3	10	7	8	10	10	10	10	19	68	+49

Under Guided Methods, conventional training methods currently in use at the ASOC and at many other TC2 units are listed first. These include classroom lectures, classroom lectures and exercises, paper-based System Training Exercises (STEs), and Computer-Assisted Instruction (CAI). Several guided forms of KEATS and SuperKEATS follow. The one most easily implemented with the current prototype systems is a guided use of KEATS and SuperKEATS where the guide would simply prescribe a simple to complex sequence of exercises or games. Another method presumes the availability of a complementary package of conventional CAI which would present instruction on decision rules and tactical object characteristics and behavior prior to dispatching the student into a KEATS exercise or a SuperKEATS game for practice. The assumption is that the student would cycle repeatedly back and forth between CAI and the KEATS or SuperKEATS practice environments in a simple to complex sequence. Finally, the category includes a full Decision Training System (DTS) which is understood as a combination of SuperKEATS and a CAI package, where the progress of the student is monitored by a student model and directed by a simple tutorial element which essentially functions as a gate to the next level of instruction.

The Cost parameters and the Instructional Effectiveness parameters are shown in Table 14 across the top. Cost ratings were assigned negative values. Instructional Effectiveness ratings were assigned as positive values. A rating of 10 was given to the relatively best or highest (cost) method on a given parameter. No rating was assigned if a parameter did not apply to a method. The sum of the C and IE ratings is a positive or negative score for each method. The scores in the rightmost column of the table provide a rough comparison of relative cost benefits between methods. The summed C and IE ratings were also plotted on an X-Y graph in Figure 11, with Instructional Effectiveness scores on the X-axis and Cost scores on the Y-axis. No relative weights were assigned to the parameters themselves because there does not appear to be a valid reference point which would allow comparison between costs and benefits in some absolute sense.

The results of this rating exercise are quite interesting. The graph shows first of all a cluster of training methods with relatively low Cost ratings and mediocre Instructional Effectiveness ratings. Among the methods in this cluster is KEATS when used in unguided or guided form. It should be noted that the methods in this cluster do not show up as more effective than FTXs or CPXs, both of which, however, are much more costly than the methods in the cluster. The graph also shows that SuperKEATS in all forms is only slightly higher in Cost ratings but significantly higher in Instructional Effectiveness than the "cluster methods."

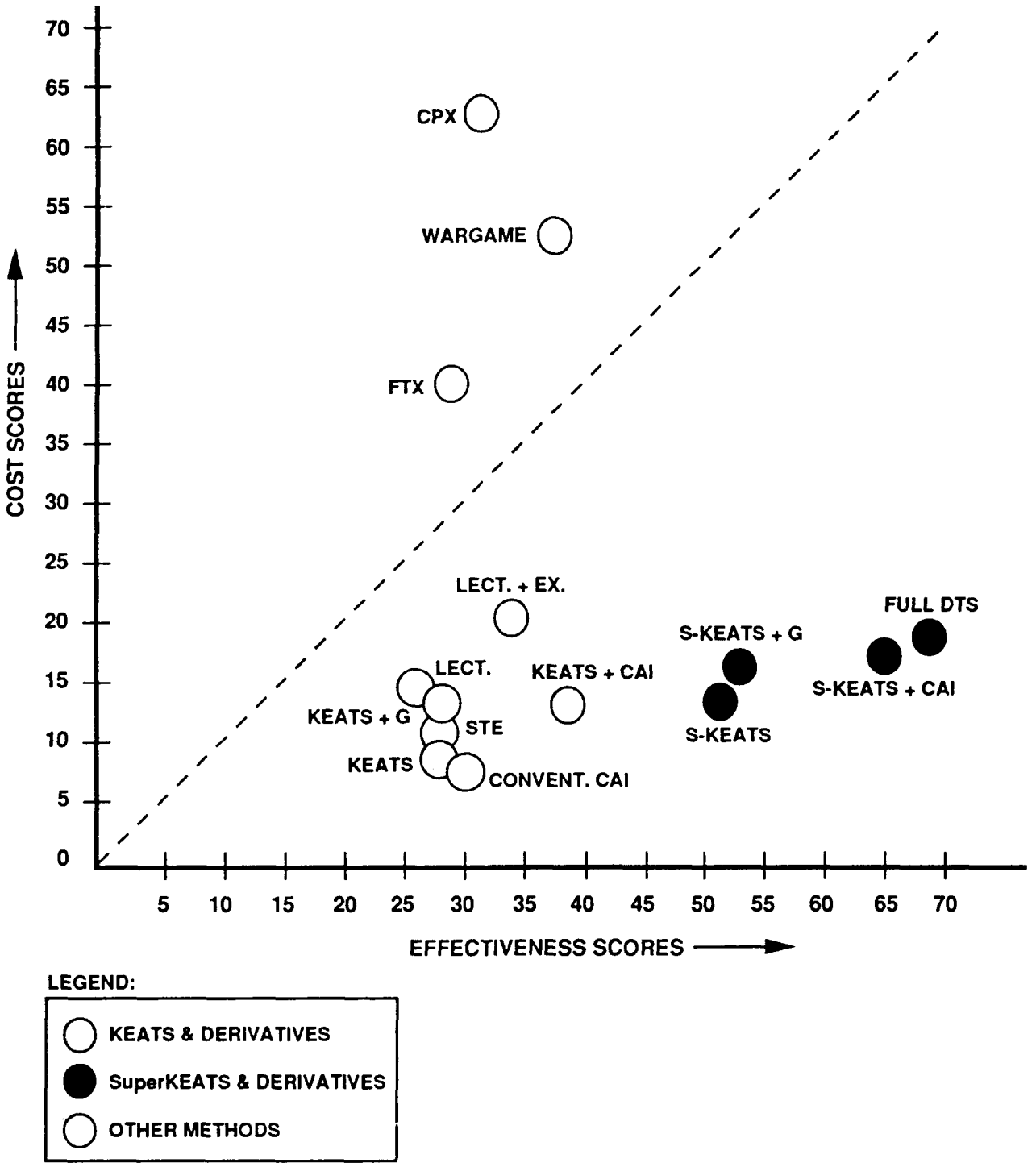


Figure 11. Cost Effectiveness Plot for 14 Methods to Train Decision-Making Skills.

Different raters would most certainly assign slightly different ratings, but it is not certain that the relative order of the methods on the 10-point scale would change. The basic outcome of this analysis, which shows a superior cost benefit ratio for the SuperKEATS training methods in any form, therefore would probably not change either.

Given the results of the cost benefit analysis described above and given the demonstrated technical feasibility of the general approach and the investment it represents, there is little doubt that the now-existing prototypes should be subjected to a rigorous program of formative and summative evaluation to determine their training effectiveness. The potential for substantial increases in training effectiveness, while keeping costs well below those of CPXs, clearly exists. If these training effectiveness increases can be verified, further investment in this technology is surely justified.

ABBREVIATIONS AND ACRONYMS

ALO	Air Liaison Officer
ARTACT	Armor Tactical Concepts Tutor
ASOC	Air Support Operations Center
ASW	Anti-Submarine Warfare
ATO	Air Tasking Order
ATOC	Allied Tactical Operations Center
BAI	Battlefield Air Interdiction
C^2	Command and Control
CAI	Computer-Aided Instruction
CPX	Command Post Exercise
DOO	Daily Operations Order
DTS	Decision Training System
EIDS	Electronic Information Delivery System
FAC	Forward Air Controller
FDO	Fighter Duty Officer
FLOT	Forward Line of Own Troops
FTX	Field Training Exercise
GEMS	Game Effectiveness Measures
ISA	Initial Situation Assessment
ISD	Instructional Systems Development
ITS	Intelligent Tutoring System
JAAT	Joint Air Attack Team
KEATS	Knowledge Engineering and Training System
POA	Plan of Action
RTB	Return to Base
SME	Subject-Matter Expert
STE	System Training Exercise
TACP	Tactical Air Control Party
TACS	Tactical Air Control System
TAOTTS	Tactical Air Operations Team Training System
TC^2	Tactical Command and Control
TEMPLAR	Tactical Expert Mission Planner
TOT	Time Over Target
UCSD	University of California at San Diego
WOC	Wing Operations Center

REFERENCES

Aagard, J.A., & Braby, R. (1976, March). Learning guidelines and algorithms for twelve types of training objectives (TAEG Report No. 23). Orlando, FL: Training Analysis and Evaluation Group, Naval Training Equipment Center.

Barnthouse, D.A. (1989). Tactical Air Operations Team Training System (TAOTTS) functional description (Unpublished manuscript).

Brecke, F. (1976). Unpublished lecture outlines for IT 555: Instructional Systems Design. Tempe, AZ: Department of Educational Technology, Arizona State University.

Brecke, F.H., Hays, P.J., Johnston, D.L., McGarvey, J.M., Peters, S.M., & Slemon, G.K. (1990). SuperKEATS: A system to support training for decision-making skills (AFHRL-TR-90-2, AD-A222 232). Wright-Patterson AFB, OH: Logistics and Human Factors Division, Air Force Human Resources Laboratory.

Brecke, F.H., Hays, P.J., Johnston, D.L., Slemon, G.K., McGarvey, J.M., & Peters, S.M. (1989). KEATS: A system to support knowledge engineering and training for decision-making skills (AFHRL-TR-89-25, AD-A216 464). Wright-Patterson AFB, OH: Logistics and Human Factors Division, Air Force Human Resources Laboratory.

Buchanan, B.G., Barstow, D., Bechtal, R., Burnett, J., Clancey, W., Kulikowski, C., Mitchell, T., & Waterman, D.A. (1984). Constructing an Expert System. In F. Hayes-Roth & D.A. Waterman (Eds.), Building Expert Systems. Reading, MA: Addison-Wesley Publishing Company.

Burton, R.R., & Brown, J.S. (1982). An investigation of computer coaching for informal learning activities. In D. Sleeman & J.S. Brown (Eds.), Intelligent Tutoring Systems. New York: Academic Press Inc.

Frank, H.G. (1969). Kybernetische Grundlagen der Paedagogik (rev. ed.). Baden-Baden, Germany: Agis-Verlag.

Fraser, B.D. (1987). Knowledge acquisition methodology (Technical Report 1094). San Diego, CA: Naval Ocean Systems Center.

Gagne, R.M., & Briggs, L.J. (1979). Principles of instructional design (2nd ed.). New York: Holt, Rinehart & Winston.

Goldstein, I. (1982). The genetic graph: A representation for the evolution of procedural knowledge. In D. Sleeman & J.S. Brown (Eds.), Intelligent Tutoring Systems. New York: Academic Press Inc.

Gropper, G.L. (1974). Instructional strategies Englewood Cliffs, NJ: Educational Technology Publications.

Hayes-Roth, F., & Waterman, D.A. (1984). An investigation of tools for building expert systems. In F. Hayes-Roth & D.A. Waterman (Eds.), Building Expert Systems. Reading, MA: Addison-Wesley Publishing Company.

Johnson, P.E. (1983). What kind of expert should a system be? The Journal of Medicine and Philosophy, 8, 77-97.

Krebs, J.C., Cream, B.W., Brecke, F., Mirman, I., Parsons, M., Silva, W., & Thorndyke, P. (1984, May). Tactical command and control combat planning and attack capability (COMPAC) Phase II (Unpublished Manuscript).

Lesgold, A., Lajoie, S., Eastman, R., Eggan, G., Gitomer, D., Glaser, R., Greenberg, L., Logan, D., Magone, M., Weiner, A., Wolf, R. & Yengo, L. (1986). Cognitive task analysis to enhance technical skills training and assessment. Air Force Human Resources Laboratory and University of Pittsburgh.

Madni, A.M., Ahlers, R., & Chu, Y. (1987, December). Knowledge-based simulation: An approach to intelligent opponent modeling for training tactical decision-making. In Proceedings for the Ninth Interservice/Industry Training Systems Conference. Washington, DC: National Security Industrial Association.

McCune, B.P. (1985). A tutorial on expert systems for battlefield applications. In Proceedings of the Seminar on Artificial Intelligence. Applications to the Battlefield. Ft. Monmouth, NJ: Armed Forces Communications and Electronics Association.

Means, B., & Gott, S.P. (1988). Cognitive task analysis as a basic for tutor development: articulating abstract knowledge representations. In J. Psotka, L.D. Massey, & S.A. Mutter (Eds.), Intelligent Tutoring Systems: Lessons Learned. Hillsdale, NJ: Lawrence Erlbaum Associates.

Merrill, M.D. (1983). In C.M. Reigeluth (Ed.), Instructional-Design Theories and Models: An Overview of their Current Status. Chapter 9, 279-333. Hillsdale, NJ: Lawrence Erlbaum Associates.

Merrill, M.D., & Wood, N.D. (1974, April). Instructional strategies: A preliminary taxonomy. Paper presented at the meeting of the Special Interest Group for Research in Mathematics Education, American Educational Research Association, Chicago, IL.

Montague, W.E. (1986, April). Application of cognitive science principles: Instructional heuristics and mechanisms for use. Paper presented at the annual meeting of the American Educational Research Association, San Francisco, CA.

Nickerson, R.S., & Feehrer, C.E. (1975). decision-making and training: A review of theoretical and empirical studies of decision-making and their implications for the training of decision makers (NAVTRAEQUIPCEN 73-C-0128-1). Cambridge, MA: Bolt Beranek and Newman, Inc.

Obermayer, R.W., Johnston, D.L., Slemon, G.S., & Hicklin, M.B. (1984). Team training: Knowledge-based simulation for team members (NAVTRAEQUIPCEN 82C-0140-1). Orlando, FL: Naval Training Equipment Center.

Parsaye, K. (1988). Acquiring & verifying knowledge automatically. AI Expert, 48-63.

Priest, P.F. (1986, May). DARPA's AirLand Battle Management Program and USAF's Tactical Expert Mission PLAnneR (TEMPLAR). In Advanced Computer Aids in the Planning and Execution of Air Warfare and Ground Strike Operations: Conference Proceedings (AD-A182 096), Meeting of the Avionics Panels of AGARD (51st), Kongsberg, Norway.

Rasmussen, J. (1986). Information processing and human-machine interaction: An approach to cognitive engineering (Series Volume 12). New York: NorthHolland.

Reigeluth, C.M. (1983). Instructional design: What is it and why is it? In C.M. Reigeluth (Ed.), Instructional-design theories and models: an overview of their current status. Hillsdale, NJ: Lawrence Erlbaum Associates.

Schraagen, J.M.C. (1986, December). Expert novice differences and their implication for knowledge elicitation techniques (Report IZF 1986-34). Kampweg 5, The Netherlands: Institute for Perception.

Stoddard, M.L., Kern, R., & Emerson, J. (1986, March). A computer-based knowledge extraction tool: A step in the development of a cognitive skills tutor. Submitted to the Association for the Development of Computer-based Instructional Systems. Alexandria, VA: U.S. Army Research Institute for Behavioral and Social Sciences.

Taylor, E.N. (1983). A review of literature relevant to unaided tactical decision-making (Research Note 83-35). Alexandria, VA: U.S. Army Research Institute for Behavioral and Social Sciences.

Waterman, D.A. (1986). A guide to expert systems. Reading, MA: Addison Wesley Publishing Company.

Wilson, J.O. (1982). Interactive microcomputer wargame for air battle (Master's thesis). Naval Postgraduate School, Monterey, CA.

* U. S. GOVERNMENT PRINTING OFFICE: 1990 - 261-951/20132